User Guide

Amazon EBS

Amazon EBS: User Guide

Table of Contents

What is Amazon Elastic Block Store?

Amazon Elastic Block Store (Amazon EBS) provides scalable, high-performance block storage resources that can be used with Amazon Elastic Compute Cloud (Amazon EC2) instances. With Amazon Elastic Block Store, you can create and manage the following block storage resources:

- **Amazon EBS volumes** — These are storage volumes that you attach to Amazon EC2 instances. After you attach a volume to an instance, you can use it in the same way you would use a local hard drive attached to a computer, for example to store files or to install applications.
- **Amazon EBS snapshots** — These are point-in-time backups of Amazon EBS volumes that persist independently from the volume itself. You can create snapshots to back up the data on your Amazon EBS volumes. You can then restore new volumes from those snapshots at any time.

Topics

Features of Amazon EBS

Amazon EBS provides the following features and benefits:

- **Multiple volume types** — Amazon EBS provides multiple volume types that allow you to optimize storage performance and cost for a broad range of applications. Volume types are divided into two major categories: **SSD-backed storage** for transactional workloads, and **HDD-backed storage** for throughput intensive workloads.
- **Scalability** — You can create Amazon EBS volumes with capacity and performance specifications that meet your needs. As your needs changes, you can use Elastic Volumes operations to dynamically increase capacity or tune performance, with no downtime.
- **Backup and recovery** — Use Amazon EBS snapshots to back up the data stored on your volumes. You can then use those snapshots to instantly restore volumes or to migrate data across AWS accounts, AWS Regions, or Availability Zones.

- **Data protection** — Use Amazon EBS encryption to encrypt your Amazon EBS volumes and Amazon EBS snapshots. Encryption operations occur on the servers that host Amazon EC2 instances, ensuring the security of both data-at-rest and data-in-transit between an instance and its attached volume and subsequent snapshots.
- **Data availability and durability** — io2 Block Express volumes provide 99.999% durability with an annual failure rate of 0.001%. Other volume types provide 99.8% to 99.9% durability with an annual failure rate of 0.1% to 0.2%. Additionally, volume data is automatically replicated across multiple servers in an Availability Zone to prevent the loss of data from the failure of any single component.
- **Data archiving** — EBS Snapshots Archive provides a low-cost storage tier to archive full, point-in-time copies of EBS Snapshots that you must retain for 90 days or more for regulatory and compliance reasons, or for future project releases.

Related services

Amazon EBS works with the following services:

- **Amazon Elastic Compute Cloud** — A service that lets you launch and manage virtual machines (Amazon EC2 instances) in the AWS Cloud. You can attach EBS volumes to those instances and use them in the same way you would use a local hard drive, for example to store files or to install applications. For more information, see What is Amazon EC2?
- **AWS Key Management Service** — A managed service that enables you to create and manage cryptographic keys. You can use AWS KMS cryptographic keys to encrypt the data stored on your Amazon EBS volumes and in your Amazon EBS snapshots. For more information, see How Amazon EBS uses AWS KMS.
- **Amazon Data Lifecycle Manager** — A managed service that automates the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs. You can use Amazon Data Lifecycle Manager to automate backups for your Amazon EBS volumes and Amazon EC2 instances. For more information, see Automate backups with Amazon Data Lifecycle Manager.
- **EBS direct APIs** — A service that enables you to create EBS snapshots, write data directly to your snapshots, read data from your snapshots, and identify the differences or changes between two snapshots. For more information, see Use EBS direct APIs to access the contents of an EBS snapshot.
- **Recycle Bin** — A data recovery service that enables you to restore accidentally deleted EBS snapshots and EBS-backed AMIs. For more information, see Recycle Bin.

Accessing Amazon EBS

You can create and manage your Amazon EBS resources using the following interfaces:

Amazon EC2 console

A web interface to create and manage volumes and snapshots. If you've signed up for an AWS account, you can access the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

AWS Command Line Interface

A command line tool that lets you manage Amazon EBS resources using commands in your command-line shell. It is supported on Windows, Mac, and Linux. For more information, see the AWS Command Line Interface User Guide and the ec2 commands.

AWS Tools for PowerShell

A set of PowerShell modules that enable you to script operations on your Amazon EBS resources from the PowerShell command line. For more information, see the AWS Tools for PowerShell User Guide and AWS Tools for PowerShell Cmdlet Reference.

CloudFormation

A fully managed AWS service that lets you create reusable JSON or YAML templates that describe your AWS resources, and then provisions and configures those resources for you. For more information, see the AWS CloudFormation User Guide.

Amazon EC2 Query API

The Amazon EC2 Query API provides HTTP or HTTPS requests that use the HTTP verb `GET` or `POST` and a query parameter named `Action`. For more information see the Amazon EC2 API Reference.

AWS SDKs

Language-specific APIs that enable you to build applications that are integrated with AWS services. AWS SDKs are available for many popular programming languages. For more information, see Tools to Build on AWS.

Pricing

With Amazon EBS, you pay only for what you provision. For more information, see Amazon EBS pricing.

Set up for Amazon EBS

Complete the tasks in this section to get set up for working with Amazon EBS resources.

Tasks

- Sign up for an AWS account
- Create a user with administrative access
- (Optional) Create and use a customer managed key for Amazon EBS encryption
- (Optional) Enable block public access for Amazon EBS snapshots

Sign up for an AWS account

If you do not have an AWS account, complete the following steps to create one.

To sign up for an AWS account

1. Open https://portal.aws.amazon.com/billing/signup.
2. Follow the online instructions.

   Part of the sign-up procedure involves receiving a phone call or text message and entering a verification code on the phone keypad.

   When you sign up for an AWS account, an *AWS account root user* is created. The root user has access to all AWS services and resources in the account. As a security best practice, assign administrative access to a user, and use only the root user to perform tasks that require root user access.

AWS sends you a confirmation email after the sign-up process is complete. At any time, you can view your current account activity and manage your account by going to https://aws.amazon.com/ and choosing **My Account**.

Create a user with administrative access

After you sign up for an AWS account, secure your AWS account root user, enable AWS IAM Identity Center, and create an administrative user so that you don't use the root user for everyday tasks.

Secure your AWS account root user

1. Sign in to the AWS Management Console as the account owner by choosing **Root user** and entering your AWS account email address. On the next page, enter your password.

   For help signing in by using root user, see Signing in as the root user in the *AWS Sign-In User Guide*.

2. Turn on multi-factor authentication (MFA) for your root user.

   For instructions, see Enable a virtual MFA device for your AWS account root user (console) in the *IAM User Guide*.

Create a user with administrative access

1. Enable IAM Identity Center.

   For instructions, see Enabling AWS IAM Identity Center in the *AWS IAM Identity Center User Guide*.

2. In IAM Identity Center, grant administrative access to a user.

   For a tutorial about using the IAM Identity Center directory as your identity source, see Configure user access with the default IAM Identity Center directory in the *AWS IAM Identity Center User Guide*.

Sign in as the user with administrative access

- To sign in with your IAM Identity Center user, use the sign-in URL that was sent to your email address when you created the IAM Identity Center user.

  For help signing in using an IAM Identity Center user, see Signing in to the AWS access portal in the *AWS Sign-In User Guide*.

Assign access to additional users

1. In IAM Identity Center, create a permission set that follows the best practice of applying least-privilege permissions.

   For instructions, see Create a permission set in the *AWS IAM Identity Center User Guide*.

2. Assign users to a group, and then assign single sign-on access to the group.

   For instructions, see Add groups in the *AWS IAM Identity Center User Guide*.

(*Optional*) Create and use a customer managed key for Amazon EBS encryption

Amazon EBS encryption is an encryption solution that uses AWS KMS cryptographic keys to encrypt your Amazon EBS volumes and Amazon EBS snapshots. Amazon EBS automatically creates a unique AWS managed KMS key for Amazon EBS encryption in each Region. This KMS key has the alias `aws/ebs`. You can't rotate the default KMS key or manage its permissions. For more flexibility and control over the KMS key used for Amazon EBS encryption, you might consider creating and using a customer managed key.

To create and use a customer managed key for Amazon EBS encryption

1. Create a symmetric encryption KMS key.
2. Select the KMS key as the default KMS key for Amazon EBS encryption.
3. Give users permission to use the KMS key for Amazon EBS encryption.

(*Optional*) Enable block public access for Amazon EBS snapshots

To prevent public sharing of your snapshots, you can enable block public access for snapshots. After you enable block public access for snapshots in a Region, any attempt to publicly share snapshots in that Region is automatically blocked. This can help you to improve the security of your snapshots and to protect your snapshot data from unauthorized or unintended access.

For more information, see Block public access for Amazon EBS snapshots.

Console

To enable block public access for snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **EC2 Dashboard**, and then in **Account attributes** (on the right-hand side), choose **Data protection and security**.

3. In the **Block public access for EBS snapshots** section, choose **Manage**.
4. Select **Block public access** and then choose one of the following options:

   - **Block all public access** — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.
   - **Block new public sharing** — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.
5. Choose **Update**.

AWS CLI

To enable block public access for snapshots

Use the enable-snapshot-block-public-access command. For `--state` specify one of the following values:

- `block-all-sharing` — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.
- `block-new-sharing` — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.

```
aws ec2 enable-snapshot-block-public-access --state block-new-sharing
```

PowerShell

To enable block public access for snapshots

Use the Enable-EC2SnapshotBlockPublicAccess cmdlet. For `-State` specify one of the following values:

- `block-all-sharing` — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.

- `block-new-sharing` — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.

```
Enable-EC2SnapshotBlockPublicAccess -State block-new-sharing
```

Amazon EBS volumes

An Amazon EBS volume is a durable, block-level storage device that you can attach to your instances. After you attach a volume to an instance, you can use it as you would use a physical hard drive. EBS volumes are flexible. For current-generation volumes attached to current-generation instance types, you can dynamically increase size, modify the provisioned IOPS capacity, and change volume type on live production volumes.

You can use EBS volumes as primary storage for data that requires frequent updates, such as the system drive for an instance or storage for a database application. You can also use them for throughput-intensive applications that perform continuous disk scans. EBS volumes persist independently from the running life of an EC2 instance.

You can attach multiple EBS volumes to a single instance. The volume and instance must be in the same Availability Zone. Depending on the volume and instance types, you can use Multi-Attach to mount a volume to multiple instances at the same time.

Amazon EBS provides the following volume types: General Purpose SSD (`gp2` and `gp3`), Provisioned IOPS SSD (`io1` and `io2`), Throughput Optimized HDD (`st1`), Cold HDD (`sc1`), and Magnetic (`standard`). They differ in performance characteristics and price, allowing you to tailor your storage performance and cost to the needs of your applications. For more information, see Amazon EBS volume types.

Your account has a limit on the total storage available to you. For more information about these limits, and how to request an increase in your limits, see Amazon EBS endpoints and quotas.

A *managed EBS volume* is managed by a service provider, such as Amazon EKS Auto Mode. You can't directly modify the settings of a managed EBS volume. Managed EBS volumes are identified by a **true** value in the **Managed** field. For more information, see Amazon EC2 managed instances.

For more information about pricing, see Amazon EBS Pricing.

Contents

- Amazon EBS volume lifecycle
- Replace an Amazon EBS volume using a snapshot
- Amazon EBS volume status checks
- Fault testing on Amazon EBS

Features and benefits of Amazon EBS volumes

EBS volumes provide benefits that are not provided by instance store volumes.

Benefits

- Data availability
- Data persistence
- Data encryption
- Data security
- Snapshots
- Flexibility

Data availability

When you create an EBS volume, it is automatically replicated within its Availability Zone to prevent data loss due to failure of any single hardware component. You can attach an EBS volume to any EC2 instance in the same Availability Zone. After you attach a volume, it appears as a native block device similar to a hard drive or other physical device. At that point, the instance can interact with the volume just as it would with a local drive. You can connect to the instance and format the EBS volume with a file system, such as `Ext4` for a Linux instance or `NTFS` for a Windows instance, and then install applications.

If you attach multiple volumes to a device that you have named, you can stripe data across the volumes for increased I/O and throughput performance.

You can attach `io1` and `io2` EBS volumes to up to 16 Nitro-based instances. For more information, see Attach an EBS volume to multiple EC2 instances using Multi-Attach. Otherwise, you can attach an EBS volume to a single instance.

You can get monitoring data for your EBS volumes, including root device volumes for EBS-backed instances, at no additional charge. For more information about monitoring metrics, see Amazon

CloudWatch metrics for Amazon EBS. For information about tracking the status of your volumes, see Amazon EventBridge events for Amazon EBS.

Data persistence

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

EBS volumes that are attached to a running instance can automatically detach from the instance with their data intact when the instance is terminated if you uncheck the **Delete on Termination** check box when you configure EBS volumes for your instance on the EC2 console. The volume can then be reattached to a new instance, enabling quick recovery. If the check box for **Delete on Termination** is checked, the volume(s) will delete upon termination of the EC2 instance. If you are using an EBS-backed instance, you can stop and restart that instance without affecting the data stored in the attached volume. The volume remains attached throughout the stop-start cycle. This enables you to process and store the data on your volume indefinitely, only using the processing and storage resources when required. The data persists on the volume until the volume is deleted explicitly. The physical block storage used by deleted EBS volumes is overwritten with zeroes or cryptographically pseudorandom data before it is allocated to a new volume. If you are dealing with sensitive data, you should consider encrypting your data manually or storing the data on a volume protected by Amazon EBS encryption. For more information, see Amazon EBS encryption.

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

By default, additional EBS volumes that are created and attached to an instance at launch are not deleted when that instance is terminated. You can modify this behavior by changing the value of the flag `DeleteOnTermination` to `true` when you launch the instance. This modified value causes the volumes to be deleted when the instance is terminated.

Data encryption

For simplified data encryption, you can create encrypted EBS volumes with the Amazon EBS encryption feature. All EBS volume types support encryption. You can use encrypted EBS volumes to meet a wide range of data-at-rest encryption requirements for regulated/audited data and applications. Amazon EBS encryption uses 256-bit Advanced Encryption Standard algorithms

(AES-256) and an Amazon-managed key infrastructure. The encryption occurs on the server that hosts the EC2 instance, providing encryption of data-in-transit from the EC2 instance to Amazon EBS storage. For more information, see Amazon EBS encryption.

Amazon EBS encryption uses AWS KMS keys when creating encrypted volumes and any snapshots created from your encrypted volumes. The first time you create an encrypted EBS volume in a Region, a default AWS managed KMS key is created for you automatically. This key is used for Amazon EBS encryption unless you create and use a customer managed key. Creating your own customer managed key gives you more flexibility, including the ability to create, rotate, disable, define access controls, and audit the encryption keys used to protect your data. For more information, see the AWS Key Management Service Developer Guide.

Data security

Amazon EBS volumes are presented to you as raw, unformatted block devices. These devices are logical devices that are created on the EBS infrastructure and the Amazon EBS service ensures that the devices are logically empty (that is, the raw blocks are zeroed or they contain cryptographically pseudorandom data) prior to any use or re-use by a customer.

If you have procedures that require that all data be erased using a specific method, either after or before use (or both), such as those detailed in **DoD 5220.22-M** (National Industrial Security Program Operating Manual) or **NIST 800-88** (Guidelines for Media Sanitization), you have the ability to do so on Amazon EBS. That block-level activity will be reflected down to the underlying storage media within the Amazon EBS service.

Snapshots

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

When you create a new volume from a snapshot, it's an exact copy of the original volume at the time the snapshot was taken. EBS volumes that are created from encrypted snapshots are automatically encrypted. By optionally specifying a different Availability Zone, you can use this functionality to create a duplicate volume in that zone. The snapshots can be shared with specific

AWS accounts or made public. When you create snapshots, you incur charges in Amazon S3 based on the size of the data being backed up, not the size of the source volume. Subsequent snapshots of the same volume are incremental snapshots. They include only changed and new data written to the volume since the last snapshot was created, and you are charged only for this changed and new data.

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

To help categorize and manage your volumes and snapshots, you can tag them with metadata of your choice.

To back up your volumes automatically, you can use Amazon Data Lifecycle Manager or AWS Backup.

Flexibility

EBS volumes support live configuration changes while in production. You can modify volume type, volume size, and IOPS capacity without service interruptions. For more information, see Modify an Amazon EBS volume using Elastic Volumes operations.

Amazon EBS volume types

Amazon EBS provides the following 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.

> **Important**
>
> There are several factors that can affect the performance of EBS volumes, such as instance configuration, I/O characteristics, and workload demand. To fully use the IOPS provisioned on an EBS volume, use EBS–optimized instances. For more information about getting the most out of your EBS volumes, see Amazon EBS volume performance.

For more information about pricing, see Amazon EBS Pricing.

Volume types

- Solid state drive (SSD) volumes
- Hard disk drive (HDD) volumes
- Previous generation volumes

Solid state drive (SSD) volumes

SSD-backed volumes are optimized for transactional workloads involving frequent read/write operations with small I/O size, where the dominant performance attribute is IOPS. SSD-backed volume types include **General Purpose SSD** and **Provisioned IOPS SSD** . The following is a summary of the use cases and characteristics of SSD-backed volumes.

	Amazon EBS General Purpose SSD volumes		Amazon EBS Provisioned IOPS SSD volumes	
Volume type	`gp3` [6]	`gp2`	`io2` Block Express	`io1`
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)		99.999% durability (0.001% annual failure rate)	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)
Use cases	• Transactional workloads • Virtual desktops • Medium-sized, single-instance databases • Low-latency interactive applications • Boot volumes • Development and test environments		Workloads that require: • Consistent sub-millisecond latency with average latency under 500 microseconds [5] • Sustained IOPS performance • More than 80,000 IOPS or	• Workloads that require sustained IOPS performance or more than 16,000 IOPS • I/O-intensive database workloads

	[Amazon EBS General Purpose SSD volumes](#)		[Amazon EBS Provisioned IOPS SSD volumes](#)	
			2,000 MiB/s of throughput	
Volume size	1 GiB - 64 TiB	1 GiB - 16 TiB	4 GiB - 64 TiB	4 GiB - 16 TiB
Max IOPS	80,000 [3] (64 KiB I/O [4])	16,000 (16 KiB I/O [4])	256,000 [3] (16 KiB I/O [4])	64,000 (16 KiB I/O [4])
Max throughput	2,000 MiB/s	250 MiB/s [1]	4,000 MiB/s	1,000 MiB/s [2]
Amazon EBS Multi-attach	Not supported		Supported	
NVMe reservations	Not supported		Supported	Not supported
Boot volume	Supported			

[1] The throughput limit is between 128 MiB/s and 250 MiB/s, depending on the volume size. For more information, see gp2 volume performance. Volumes created before **December 3, 2018** that have not been modified since creation might not reach full performance unless you modify the volume.

[2] To achieve maximum throughput of 1,000 MiB/s, the volume must be provisioned with 64,000 IOPS and it must be attached to a Nitro-based instance. Volumes created before **December 6, 2017** that have not been modified since creation might not reach full performance unless you modify the volume.

[3] Nitro-based instances support volumes provisioned with up to 256,000 IOPS. Other instance types can be attached to volumes provisioned with up to 64,000 IOPS, but can achieve up to 32,000 IOPS.

[4] Represents the required I/O size to reach maximum IOPS within the volume's throughput limit.

[5] `io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations.

[6] On Outposts, gp3 volumes support sizes up to 16 TiB, IOPS up to 16,000, and throughput up to 1,000 MiB/s.

For more information about the SSD-backed volume types, see the following:

- Amazon EBS General Purpose SSD volumes
- Amazon EBS Provisioned IOPS SSD volumes

Hard disk drive (HDD) volumes

HDD-backed volumes are optimized for large streaming workloads where the dominant performance attribute is throughput. HDD volume types include **Throughput Optimized HDD** and **Cold HDD**. The following is a summary of the use cases and characteristics of HDD-backed volumes.

	Throughput Optimized HDD volumes	**Cold HDD volumes**
Volume type	`st1`	`sc1`
Durability	99.8% - 99.9% durability (0.1% - 0.2% annual failure rate)	
Use cases	• Big data • Data warehouses • Log processing	• Throughput-oriented storage for data that is infrequently accessed • Scenarios where the lowest storage cost is important
Volume size	125 GiB - 16 TiB	

	Throughput Optimized HDD volumes	Cold HDD volumes
Max IOPS per volume (1 MiB I/O)	500	250
Max throughput per volume	500 MiB/s	250 MiB/s
Amazon EBS Multi-attach	Not supported	
Boot volume	Not supported	

For more information about the Hard disk drives (HDD) volumes, see Amazon EBS Throughput Optimized HDD and Cold HDD volumes.

Previous generation volumes

Magnetic (`standard`) volumes are previous generation volumes that are backed by magnetic drives. They are suited for workloads with small datasets where data is accessed infrequently and performance is not of primary importance. These volumes deliver approximately 100 IOPS on average, with burst capability of up to hundreds of IOPS, and they can range in size from 1 GiB to 1 TiB.

> **Tip**
>
> Magnetic is a previous generation volume type. If you need higher performance or performance consistency than previous-generation volumes can provide, we recommend using one of the current generation volume types.

The following table describes previous-generation EBS volume types.

	Magnetic
Volume type	`standard`

	Magnetic
Use cases	Workloads where data is infrequently accessed
Volume size	1 GiB-1 TiB
Max IOPS per volume	40–200
Max throughput per volume	40–90 MiB/s
Boot volume	Supported

Amazon EBS General Purpose SSD volumes

General Purpose SSD (gp2 and gp3) volumes are backed by solid-state drives (SSDs). They balance price and performance for a wide variety of transactional workloads. These include virtual desktops, medium-sized single instance databases, latency sensitive interactive applications, development and test environments, and boot volumes. We recommend these volumes for most workloads.

Amazon EBS offers the following types of General Purpose SSD volumes:

Types

- General Purpose SSD (gp3) volumes
- General Purpose SSD (gp2) volumes

General Purpose SSD (gp3) volumes

General Purpose SSD (gp3) volumes are the latest generation of General Purpose SSD volumes, and the lowest cost SSD volume offered by Amazon EBS. This volume type helps to provide the right balance of price and performance for most applications. It also helps you to scale volume performance independently of volume size. This means that you can provision the required performance without needing to provision additional block storage capacity. Additionally, gp3 volumes offer a 20 percent lower price per GiB than General Purpose SSD (gp2) volumes.

gp3 volumes provide single-digit millisecond latency and 99.8 percent to 99.9 percent volume durability with an annual failure rate (AFR) no higher than 0.2 percent, which translates to a maximum of two volume failures per 1,000 running volumes over a one-year period. AWS designs gp3 volumes to deliver their provisioned performance 99 percent of the time.

> **Tip**
>
> For latency-sensitive workloads, we recommend that you use io2 Block Express volumes. `io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations. `io2` Block Express volumes also deliver better outlier latency compared to General Purpose volumes, reducing the frequency of I/Os exceeding 800 microseconds by over 10 times. For more information, see Provisioned IOPS SSD (`io2`) Block Express volumes.

Contents

gp3 volume performance

> **Tip**
>
> gp3 volumes do not use burst performance. They can indefinitely sustain their full provisioned IOPS and throughput performance.

IOPS performance

gp3 volumes deliver a consistent baseline IOPS performance of 3,000 IOPS, which is included with the price of storage. You can provision additional IOPS (up to a maximum of 80,000) for an additional cost at a ratio of 500 IOPS per GiB of volume size. Maximum IOPS can be provisioned for volumes 160 GiB or larger (500 IOPS per GiB × 160 GiB = 80,000 IOPS).

Throughput performance

gp3 volumes deliver a consistent baseline throughput performance of 125 MiB/s, which is included with the price of storage. You can provision additional throughput (up to a maximum of 2,000 MiB/

s) for an additional cost at a ratio of 0.25 MiB/s per provisioned IOPS. Maximum throughput can be provisioned at 8,000 IOPS or higher and 16 GiB or larger (8,000 IOPS × 0.25 MiB/s per IOPS = 2,000 MiB/s).

> **Note**
>
> On Outposts, gp3 volumes support sizes up to 16 TiB, IOPS up to 16,000, and throughput up to 1,000 MiB/s.

gp3 volume size

A gp3 volume can range in size from 1 GiB to 64 TiB.

Migrate to gp3 from gp2

If you are currently using gp2 volumes, you can migrate your volumes to gp3 using Modify an Amazon EBS volume using Elastic Volumes operations operations. You can use Amazon EBS Elastic Volumes operations to modify the volume type, IOPS, and throughput of your existing volumes without interrupting your Amazon EC2 instances. When using the console to create a volume or to create an AMI from a snapshot, General Purpose SSD gp3 is the default selection for volume type. In other cases, gp2 is the default selection. In these cases, you can select gp3 as the volume type instead of using gp2.

To find out how much you can save by migrating your gp2 volumes to gp3, use the Amazon EBS gp2 to gp3 migration cost savings calculator.

General Purpose SSD (gp2) volumes

They offer cost-effective storage that is ideal for a broad range of transactional workloads. With gp2 volumes, performance scales with volume size.

> **Tip**
>
> gp3 volumes are the latest generation of General Purpose SSD volumes. They offer more predictable performance scaling and prices that are up to 20 percent lower than gp2 volumes. For more information, see General Purpose SSD (gp3) volumes.
> To find out how much you can save by migrating your gp2 volumes to gp3, use the Amazon EBS gp2 to gp3 migration cost savings calculator.

gp2 volumes provide single-digit millisecond latency and 99.8 percent to 99.9 percent volume durability with an annual failure rate (AFR) no higher than 0.2 percent, which translates to a maximum of two volume failures per 1,000 running volumes over a one-year period. AWS designs gp2 volumes to deliver their provisioned performance 99 percent of the time.

Contents

gp2 volume performance

IOPS performance

Baseline IOPS performance scales linearly between a minimum of 100 and a maximum of 16,000 at a rate of 3 IOPS per GiB of volume size. IOPS performance is provisioned as follows:

- Volumes 33.33 GiB and smaller are provisioned with the minimum of 100 IOPS.
- Volumes larger than 33.33 GiB are provisioned with 3 IOPS per GiB of volume size up to the maximum of 16,000 IOPS, which is reached at 5,334 GiB (3 X 5,334).
- Volumes 5,334 GiB and larger are provisioned with 16,000 IOPS.

gp2 volumes smaller than 1 TiB (and that are provisioned with less than 3,000 IOPS) can **burst** to 3,000 IOPS when needed for an extended period of time. A volume's ability to burst is governed by I/O credits. When I/O demand is greater than baseline performance, the volume **spends I/O credits** to burst to the required performance level (up to 3,000 IOPS). While bursting, I/O credits are not accumulated and they are spent at the rate of IOPS that is being used above baseline IOPS (spend rate = burst IOPS - baseline IOPS). The more I/O credits a volume has accrued, the longer it can sustain its burst performance. You can calculate **Burst duration** as follows:

```
                        (I/O credit balance)
Burst duration  =  ------------------------------
                   (Burst IOPS) - (Baseline IOPS)
```

When I/O demand drops to baseline performance level or lower, the volume starts to **earn I/O credits** at a rate of 3 I/O credits per GiB of volume size per second. Volumes have an **I/O credit accrual limit** of 5.4 million I/O credits, which is enough to sustain the maximum burst performance of 3,000 IOPS for at least 30 minutes.

Note

Each volume receives an initial I/O credit balance of 5.4 million I/O credits, which provides a fast initial boot cycle for boot volumes and a good bootstrapping experience for other applications.

The following table lists example volume sizes and the associated baseline performance of the volume, the burst duration (when starting with 5.4 million I/O credits), and the time needed to refill an empty I/O credits balance.

Volume size (GiB)	Baseline performance (IOPS)	Burst duration at 3,000 IOPS (seconds)	Time to refill empty credit balance (seconds)
1 to 33.33	100	1,862	54,000
100	300	2,000	18,000
334 (min size for max throughput)	1,002	2,703	5,389
750	2,250	7,200	2,400
1,000	3,000	N/A*	N/A*
5,334 (min size for max IOPS) and larger	16,000	N/A*	N/A*

* The baseline performance of the volume exceeds the maximum burst performance.

You can monitor the I/O credit balance for a volume using the Amazon EBS `BurstBalance` metric in Amazon CloudWatch. This metric shows the percentage of I/O credits for `gp2` remaining. For more information, see [Amazon EBS I/O characteristics and monitoring](). You can set an alarm that notifies you when the `BurstBalance` value falls to a certain level. For more information, see [Creating CloudWatch Alarms]().

Throughput performance

gp2 volumes deliver throughput between 128 MiB/s and 250 MiB/s, depending on the volume size. Throughput performance is provisioned as follows:

- Volumes that are 170 GiB and smaller deliver a maximum throughput of 128 MiB/s.
- Volumes larger than 170 GiB but smaller than 334 GiB can burst to a maximum throughput of 250 MiB/s.
- Volumes that are 334 GiB and larger deliver 250 MiB/s.

Throughput for a gp2 volume can be calculated using the following formula, up to the throughput limit of 250 MiB/s:

```
Throughput in MiB/s = IOPS performance × I/O size in KiB / 1,024
```

gp2 volume size

A gp2 volume can range in size from 1 GiB to 16 TiB. Keep in mind that volume performance scales linearly with the volume size.

Amazon EBS Provisioned IOPS SSD volumes

Provisioned IOPS SSD volumes are backed by solid-state drives (SSDs). They are the highest performance Amazon EBS storage volumes designed for critical, IOPS-intensive, and throughput-intensive workloads that require low latency. Provisioned IOPS SSD volumes deliver their provisioned IOPS performance 99.9 percent of the time.

Amazon EBS offers two types of Provisioned IOPS SSD volumes:

- Provisioned IOPS SSD (io2) Block Express volumes
- Provisioned IOPS SSD (io1) volumes

Provisioned IOPS SSD (io2) Block Express volumes

io2 Block Express volumes are built on the next generation of Amazon EBS storage server architecture. It has been built for the purpose of meeting the performance requirements of the most demanding I/O intensive applications that run on instances built on the Nitro System. With the highest durability and lowest latency, Block Express is ideal for running performance-intensive, mission-critical workloads, such as Oracle, SAP HANA, Microsoft SQL Server, and SAS Analytics.

Block Express architecture increases performance and scale of `io2` volumes. Block Express servers communicate with [Nitro-based instances] using the Scalable Reliable Datagram (SRD) networking protocol. This interface is implemented in the Nitro Card dedicated for Amazon EBS I/O function on the host hardware of the instance. It minimizes I/O delay and latency variation (network jitter), which provides faster and more consistent performance for your applications.

`io2` Block Express volumes are designed to provide 99.999 percent volume durability with an annual failure rate (AFR) no higher than 0.001 percent, which translates to a single volume failure per 100,000 running volumes over a one-year period. `io2` Block Express volumes are suited for workloads that benefit from a single volume that provides consistent sub-millisecond latency, and supports higher IOPS and throughput than gp3 volumes.

When attached to Nitro-based instances, `io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations. `io2` Block Express volumes also deliver better outlier latency compared to General Purpose volumes, reducing the frequency of I/Os exceeding 800 microseconds by over 10 times.

Topics

- Considerations
- Performance

Considerations

- `io2` Block Express volumes are available in all AWS Regions, including the AWS GovCloud (US) Regions and China Regions.
- As of **April 30, 2025**, all new and previously created `io2` volumes are `io2` Block Express volumes.
- Nitro-based instances support volumes provisioned with up to 256,000 IOPS. Other instance types can be attached to volumes provisioned with up to 64,000 IOPS, but can achieve up to 32,000 IOPS.

Performance

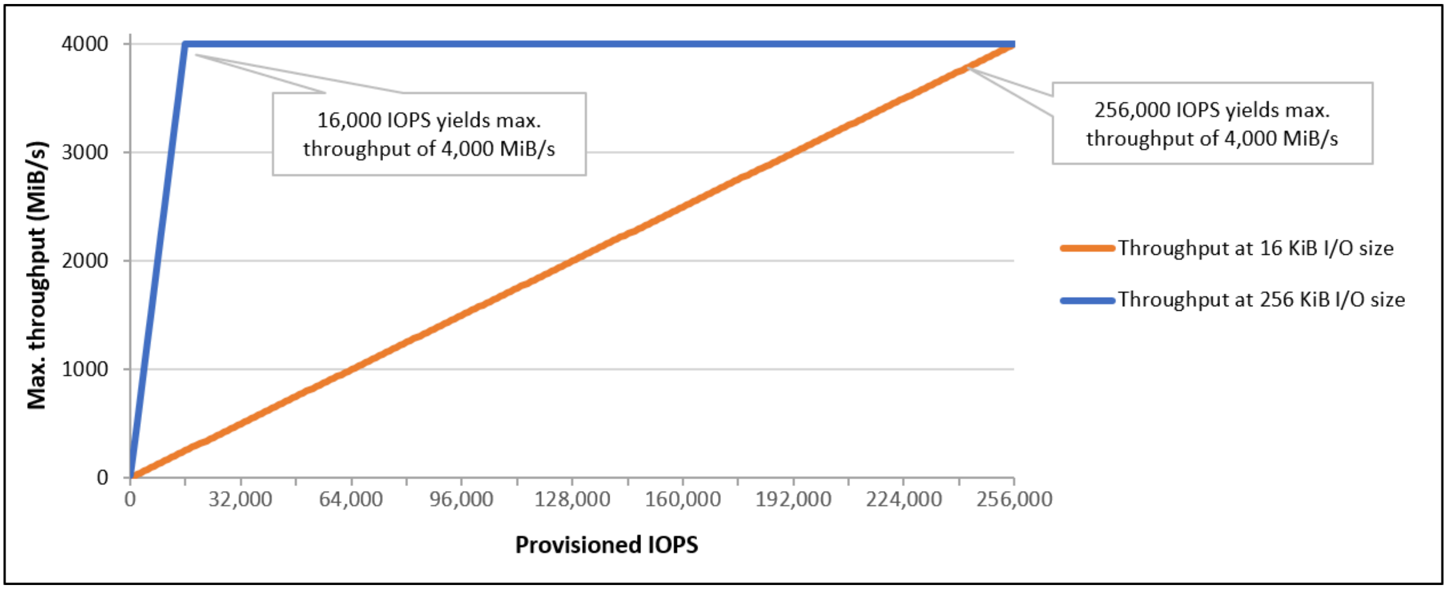
`io2` Block Express volumes have the following characteristics:

- Average latency under 500 microseconds for 16KiB I/O size. Better outlier latency compared to General Purpose volumes, reducing the frequency of I/Os exceeding 800 microseconds by over 10 times.

- Storage capacity up to 64 TiB (65,536 GiB).
- Provisioned IOPS up to 256,000, with an IOPS:GiB ratio of 1,000:1. Maximum IOPS can be provisioned with volumes 256 GiB and larger (1,000 IOPS × 256 GiB = 256,000 IOPS).

> **Note**
>
> You can achieve up to 256,000 IOPS with Nitro-based instances. On other instances, you can achieve up to 32,000 IOPS.

- Volume throughput up to 4,000 MiB/s. Throughput scales proportionally at a rate of 0.256 MiB/s per provisioned IOPS. Maximum throughput can be achieved at 16,000 IOPS or higher.



Provisioned IOPS SSD (io1) volumes

Provisioned IOPS SSD (`io1`) volumes are designed to meet the needs of I/O-intensive workloads, particularly database workloads, that are sensitive to storage performance and consistency. Provisioned IOPS SSD volumes use a consistent IOPS rate, which you specify when you create the volume, and Amazon EBS delivers the provisioned performance 99.9 percent of the time.

`io1` volumes are designed to provide 99.8 percent to 99.9 percent volume durability with an annual failure rate (AFR) no higher than 0.2 percent, which translates to a maximum of two volume failures per 1,000 running volumes over a one-year period.

`io1` volumes are available for all Amazon EC2 instance types.

Performance

`io1` volumes can range in size from 4 GiB to 16 TiB and you can provision from 100 IOPS up to 64,000 IOPS per volume. The maximum ratio of provisioned IOPS to requested volume size (in GiB) is 50:1. For example, a 100 GiB `io1` volume can be provisioned with up to 5,000 IOPS.

The maximum IOPS can be provisioned for volumes that are 1,280 GiB or larger (50 × 1,280 GiB = 64,000 IOPS).

- `io1` volumes provisioned with up to 32,000 IOPS support a maximum I/O size of 256 KiB and yield as much as 500 MiB/s of throughput. With the I/O size at the maximum, peak throughput is reached at 2,000 IOPS.
- `io1` volumes provisioned with more than 32,000 IOPS (up to the maximum of 64,000 IOPS) yield a linear increase in throughput at a rate of 16 KiB per provisioned IOPS. For example, a volume provisioned with 48,000 IOPS can support up to 750 MiB/s of throughput (16 KiB per provisioned IOPS × 48,000 provisioned IOPS = 750 MiB/s).
- To achieve the maximum throughput of 1,000 MiB/s, a volume must be provisioned with 64,000 IOPS (16 KiB per provisioned IOPS × 64,000 provisioned IOPS = 1,000 MiB/s).
- You can achieve up to 64,000 IOPS only on [Nitro-based instances](). On other instances, you can achieve up to 32,000 IOPS.

. The following graph illustrates these performance characteristics:

Your per-I/O latency experience depends on the provisioned IOPS and on your workload profile. For the best I/O latency experience, ensure that you provision IOPS to meet the I/O profile of your workload.

Amazon EBS Throughput Optimized HDD and Cold HDD volumes

The HDD-backed volumes provided by Amazon EBS fall into these categories:

- Throughput Optimized HDD — A low-cost HDD designed for frequently accessed, throughput-intensive workloads.
- Cold HDD — The lowest-cost HDD designed for less frequently accessed workloads.

Topics

Limitations on per-instance throughput

Throughput for `st1` and `sc1` volumes is always determined by the smaller of the following:

- Throughput limits of the volume
- Throughput limits of the instance

As for all Amazon EBS volumes, we recommend that you select an appropriate EBS-optimized EC2 instance to avoid network bottlenecks.

Throughput Optimized HDD volumes

Throughput Optimized HDD (`st1`) volumes provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. This volume type is a good fit for large, sequential workloads such as Amazon EMR, ETL, data warehouses, and log processing. Bootable `st1` volumes are not supported.

Throughput Optimized HDD (`st1`) volumes, though similar to Cold HDD (`sc1`) volumes, are designed to support *frequently* accessed data.

Note

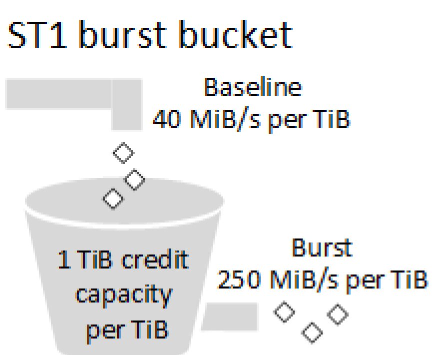
This volume type is optimized for workloads involving large, sequential I/O, and we recommend that customers with workloads performing small, random I/O use Amazon EBS General Purpose SSD volumes or Amazon EBS Provisioned IOPS SSD volumes. For more information, see Inefficiency of small read/writes on HDD.

Throughput Optimized HDD (`st1`) volumes attached to EBS-optimized instances are designed to offer consistent performance, delivering at least 90 percent of the expected throughput performance 99 percent of the time in a given year.

Throughput credits and burst performance

Like `gp2`, `st1` uses a burst bucket model for performance. Volume size determines the baseline throughput of your volume, which is the rate at which the volume accumulates throughput credits. Volume size also determines the burst throughput of your volume, which is the rate at which you can spend credits when they are available. Larger volumes have higher baseline and burst throughput. The more credits your volume has, the longer it can drive I/O at the burst level.

The following diagram shows the burst bucket behavior for `st1`.



Subject to throughput and throughput-credit caps, the available throughput of an `st1` volume is expressed by the following formula:

```
(Volume size) × (Credit accumulation rate per TiB) = Throughput
```

For a 1-TiB `st1` volume, burst throughput is limited to 250 MiB/s, the bucket fills with credits at 40 MiB/s, and it can hold up to 1 TiB-worth of credits.

Larger volumes scale these limits linearly, with throughput capped at a maximum of 500 MiB/s. After the bucket is depleted, throughput is limited to the baseline rate of 40 MiB/s per TiB.

On volume sizes ranging from 0.125 TiB to 16 TiB, baseline throughput varies from 5 MiB/s to a cap of 500 MiB/s, which is reached at 12.5 TiB as follows:

```
             40 MiB/s
12.5 TiB × ---------- = 500 MiB/s
              1 TiB
```

Burst throughput varies from 31 MiB/s to a cap of 500 MiB/s, which is reached at 2 TiB as follows:

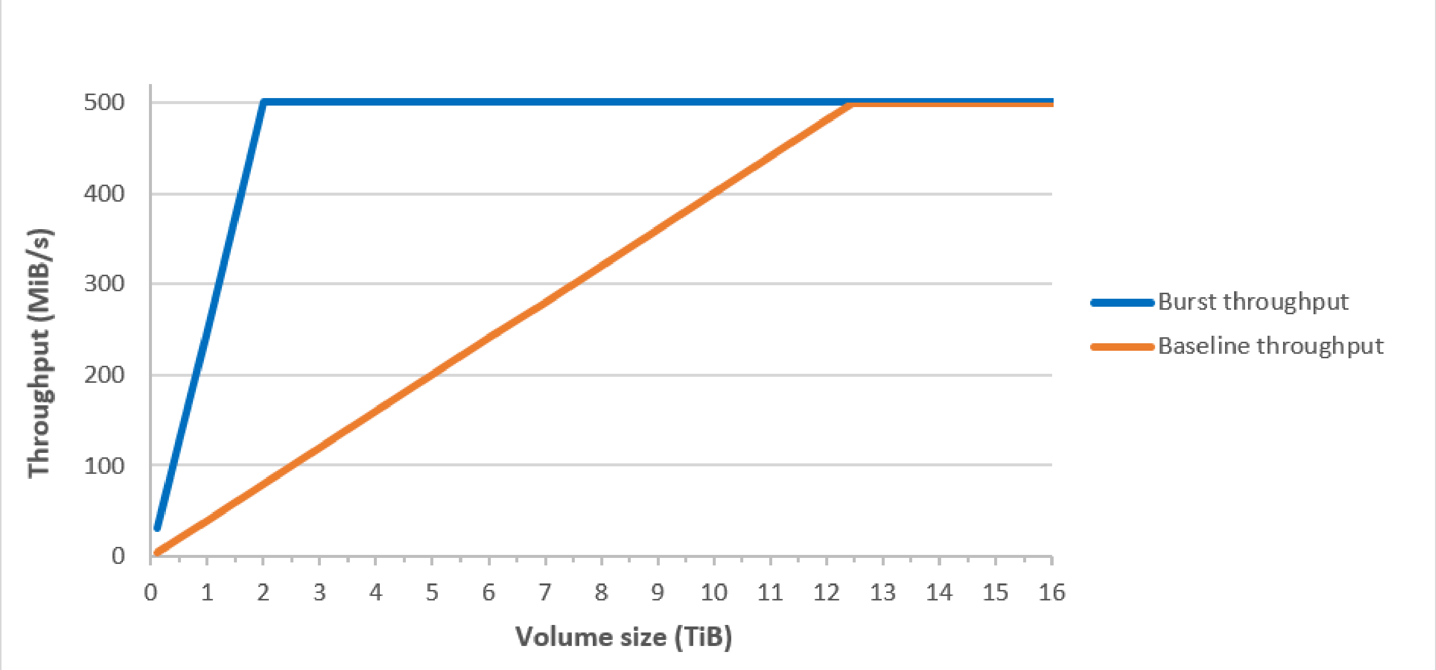
```
          250 MiB/s
2 TiB × ---------- = 500 MiB/s
           1 TiB
```

The following table states the full range of base and burst throughput values for `st1`.

Volume size (TiB)	ST1 base throughput (MiB/s)	ST1 burst throughput (MiB/s)
0.125	5	31
0.5	20	125
1	40	250
2	80	500
3	120	500
4	160	500
5	200	500
6	240	500
7	280	500
8	320	500

Volume size (TiB)	ST1 base throughput (MiB/s)	ST1 burst throughput (MiB/s)
9	360	500
10	400	500
11	440	500
12	480	500
12.5	500	500
13	500	500
14	500	500
15	500	500
16	500	500

The following diagram plots the table values:



> **Note**
>
> When you create a snapshot of a Throughput Optimized HDD (`st1`) volume, performance may drop as far as the volume's baseline value while the snapshot is in progress.

For information about using CloudWatch metrics and alarms to monitor your burst bucket balance, see Monitor the burst bucket balance for volumes.

Cold HDD volumes

Cold HDD (`sc1`) volumes provide low-cost magnetic storage that defines performance in terms of throughput rather than IOPS. With a lower throughput limit than `st1`, `sc1` is a good fit for large, sequential cold-data workloads. If you require infrequent access to your data and are looking to save costs, `sc1` provides inexpensive block storage. Bootable `sc1` volumes are not supported.

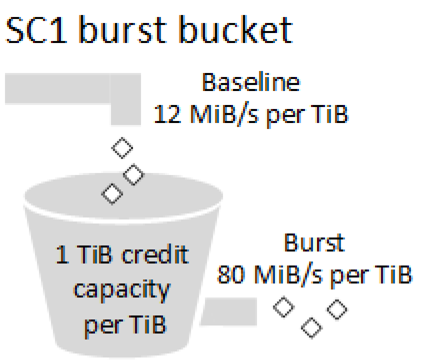
Cold HDD (`sc1`) volumes, though similar to Throughput Optimized HDD (`st1`) volumes, are designed to support *infrequently* accessed data.

> **Note**
>
> This volume type is optimized for workloads involving large, sequential I/O, and we recommend that customers with workloads performing small, random I/O use Amazon EBS General Purpose SSD volumes or Amazon EBS Provisioned IOPS SSD volumes. For more information, see Inefficiency of small read/writes on HDD.

Cold HDD (`sc1`) volumes attached to EBS-optimized instances are designed to offer consistent performance, delivering at least 90 percent of the expected throughput performance 99 percent of the time in a given year.

Throughput credits and burst performance

Like `gp2`, `sc1` uses a burst bucket model for performance. Volume size determines the baseline throughput of your volume, which is the rate at which the volume accumulates throughput credits. Volume size also determines the burst throughput of your volume, which is the rate at which you can spend credits when they are available. Larger volumes have higher baseline and burst throughput. The more credits your volume has, the longer it can drive I/O at the burst level.



Subject to throughput and throughput-credit caps, the available throughput of an `sc1` volume is expressed by the following formula:

```
(Volume size) × (Credit accumulation rate per TiB) = Throughput
```

For a 1-TiB `sc1` volume, burst throughput is limited to 80 MiB/s, the bucket fills with credits at 12 MiB/s, and it can hold up to 1 TiB-worth of credits.

Larger volumes scale these limits linearly, with throughput capped at a maximum of 250 MiB/s. After the bucket is depleted, throughput is limited to the baseline rate of 12 MiB/s per TiB.

On volume sizes ranging from 0.125 TiB to 16 TiB, baseline throughput varies from 1.5 MiB/s to a maximum of 192 MiB/s, which is reached at 16 TiB as follows:

```
           12 MiB/s
16 TiB × ---------- = 192 MiB/s
            1 TiB
```

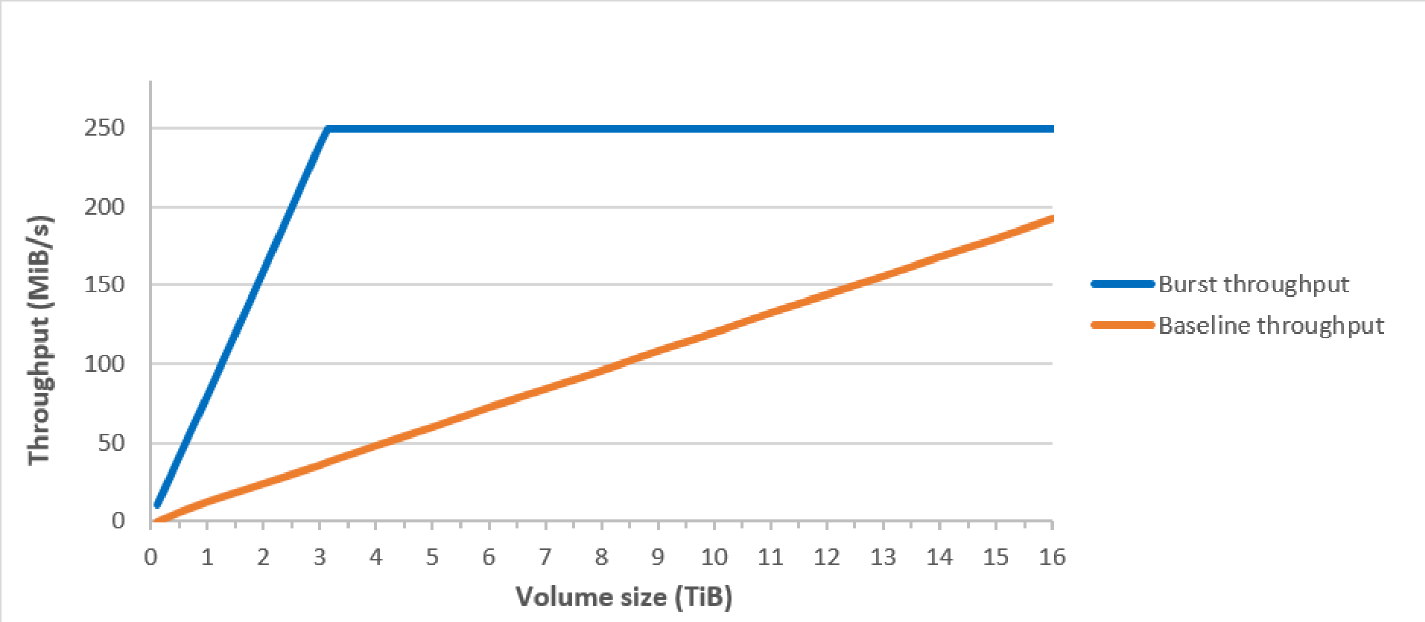
Burst throughput varies from 10 MiB/s to a cap of 250 MiB/s, which is reached at 3.125 TiB as follows:

```
             80 MiB/s
3.125 TiB × ----------- = 250 MiB/s
              1 TiB
```

The following table states the full range of base and burst throughput values for `sc1`:

Volume Size (TiB)	SC1 Base Throughput (MiB/s)	SC1 Burst Throughput (MiB/s)
0.125	1.5	10
0.5	6	40
1	12	80
2	24	160
3	36	240
3.125	37.5	250
4	48	250
5	60	250
6	72	250
7	84	250
8	96	250
9	108	250
10	120	250
11	132	250
12	144	250
13	156	250
14	168	250
15	180	250
16	192	250

The following diagram plots the table values:



> **Note**
>
> When you create a snapshot of a Cold HDD (`sc1`) volume, performance may drop as far as the volume's baseline value while the snapshot is in progress.

For information about using CloudWatch metrics and alarms to monitor your burst bucket balance, see Monitor the burst bucket balance for volumes.

Performance considerations when using HDD volumes

For optimal throughput results using HDD volumes, plan your workloads with the following considerations in mind.

Comparing Throughput Optimized HDD and Cold HDD

The `st1` and `sc1` bucket sizes vary according to volume size, and a full bucket contains enough tokens for a full volume scan. However, larger `st1` and `sc1` volumes take longer for the volume scan to complete because of per-instance and per-volume throughput limits. Volumes attached to smaller instances are limited to the per-instance throughput rather than the `st1` or `sc1` throughput limits.

Both `st1` and `sc1` are designed for performance consistency of 90 percent of burst throughput 99 percent of the time. Non-compliant periods are approximately uniformly distributed, targeting 99 percent of expected total throughput each hour.

In general, scan times are expressed by this formula:

```
 Volume size
------------ = Scan time
 Throughput
```

For example, taking the performance consistency guarantees and other optimizations into account, an `st1` customer with a 5-TiB volume can expect to complete a full volume scan in 2.91 to 3.27 hours.

- Optimal scan time

```
   5 TiB            5 TiB
----------- = ------------------ = 10,486 seconds = 2.91 hours
 500 MiB/s     0.00047684 TiB/s
```

- Maximum scan time

```
  2.91 hours
-------------- = 3.27 hours
 (0.90)(0.99) <-- From expected performance of 90% of burst 99% of the time
```

Similarly, an `sc1` customer with a 5-TiB volume can expect to complete a full volume scan in 5.83 to 6.54 hours.

- Optimal scan time

```


   5 TiB             5 TiB
----------- = ------------------- = 20972 seconds = 5.83 hours
 250 MiB/s     0.000238418 TiB/s
```

- Maximum scan time

```
  5.83 hours
-------------- = 6.54 hours
```

```
(0.90)(0.99)
```

The following table shows ideal scan times for volumes of various size, assuming full buckets and sufficient instance throughput.

Volume size (TiB)	ST1 scan time with burst (hours)*	SC1 scan time with burst (hours)*
1	1.17	3.64
2	1.17	3.64
3	1.75	3.64
4	2.33	4.66
5	2.91	5.83
6	3.50	6.99
7	4.08	8.16
8	4.66	9.32
9	5.24	10.49
10	5.83	11.65
11	6.41	12.82
12	6.99	13.98
13	7.57	15.15
14	8.16	16.31
15	8.74	17.48
16	9.32	18.64

* These scan times assume an average queue depth (rounded to the nearest whole number) of four or more when performing 1 MiB of sequential I/O.

Therefore if you have a throughput-oriented workload that needs to complete scans quickly (up to 500 MiB/s), or requires several full volume scans a day, use `st1`. If you are optimizing for cost, your data is relatively infrequently accessed, and you don't need more than 250 MiB/s of scanning performance, then use `sc1`.

Inefficiency of small read/writes on HDD

The performance model for `st1` and `sc1` volumes is optimized for sequential I/Os, favoring high-throughput workloads, offering acceptable performance on workloads with mixed IOPS and throughput, and discouraging workloads with small, random I/O.

For example, an I/O request of 1 MiB or less counts as a 1 MiB I/O credit. However, if the I/Os are sequential, they are merged into 1 MiB I/O blocks and count only as a 1 MiB I/O credit.

Monitor the burst bucket balance for volumes

You can monitor the burst bucket level for `st1` and `sc1` volumes using the Amazon EBS `BurstBalance` metric available in Amazon CloudWatch. This metric shows the throughput credits for `st1` and `sc1` remaining in the burst bucket. For more information about the `BurstBalance` metric and other metrics related to I/O, see [Amazon EBS I/O characteristics and monitoring](). CloudWatch also allows you to set an alarm that notifies you when the `BurstBalance` value falls to a certain level. For more information, see [Creating CloudWatch Alarms]().

Amazon EBS volume constraints

The size of an Amazon EBS volume is constrained by the physics and arithmetic of block data storage, as well as by the implementation decisions of operating system (OS) and file system designers. AWS imposes additional limits on volume size to safeguard the reliability of its services.

The following sections describe the most important factors that limit the usable size of an EBS volume and offer recommendations for configuring your EBS volumes.

Contents

- Data block sizes

Storage capacity

The following table summarizes the theoretical and implemented storage capacities for the most commonly used file systems on Amazon EBS, assuming a 4,096 byte block size.

Partitioning scheme	Max addressable blocks	Theoretical max size (blocks × block size)	Ext4 implemented max size*	XFS implemented max size**	NTFS implemented max size	Max supported by EBS
MBR	2^{32}	2 TiB	2 TiB	2 TiB	2 TiB	2 TiB
GPT	2^{64}	64 ZiB	1 EiB = 1024^2 TiB (50 TiB certified on RHEL7)	500 TiB (certified on RHEL7)	256 TiB	64 TiB †

* Ext4 Howto and What are the file and system size limits for Red Hat Enterprise Linux?

** What are the file and system size limits for Red Hat Enterprise Linux?

† `io2` Block Express volumes support up to 64 TiB for GPT partitions. For more information, see Provisioned IOPS SSD (`io2`) Block Express volumes.

Service limitations

Amazon EBS abstracts the massively distributed storage of a data center into virtual hard disk drives. To an operating system installed on an EC2 instance, an attached EBS volume appears to be a physical hard disk drive containing 512-byte disk sectors. The OS manages the allocation of data blocks (or clusters) onto those virtual sectors through its storage management utilities. The allocation is in conformity with a volume partitioning scheme, such as master boot record (MBR) or GUID partition table (GPT), and within the capabilities of the installed file system (ext4, NTFS, and so on).

EBS is not aware of the data contained in its virtual disk sectors; it only ensures the integrity of the sectors. This means that AWS actions and OS actions are independent of each other. When you are selecting a volume size, be aware of the capabilities and limits of both, as in the following cases:

- EBS currently supports a maximum volume size of 64 TiB. This means that you can create an EBS volume as large as 64 TiB, but whether the OS recognizes all of that capacity depends on its own design characteristics and on how the volume is partitioned.
- Boot volumes must use either the MBR or GPT partitioning scheme. The AMI you launch an instance from determines the boot mode and subsequently the partition scheme used for the boot volume.

  With **MBR**, boot volumes are limited to 2 TiB in size.

  With **GPT**, boot volumes can be up to 64 TiB in size when used with GRUB2 (Linux) or UEFI boot mode (Windows).

  For more information, see Make an Amazon EBS volume available for use.
- Non-boot volumes that are 2 TiB (2048 GiB) or larger must use a GPT partition table to access the entire volume.

Partitioning schemes

Among other impacts, the partitioning scheme determines how many logical data blocks can be uniquely addressed in a single volume. For more information, see Data block sizes. The common partitioning schemes in use are *Master Boot Record* (MBR) and *GUID partition table* (GPT). The important differences between these schemes can be summarized as follows.

MBR

MBR uses a 32-bit data structure to store block addresses. This means that each data block is mapped with one of 2^{32} possible integers. The maximum addressable size of a volume is given by the following formula:

$$2^{32} \times \text{Block size}$$

The block size for MBR volumes is conventionally limited to 512 bytes. Therefore:

$$2^{32} \times 512 \text{ bytes} = 2 \text{ TiB}$$

Engineering workarounds to increase this 2-TiB limit for MBR volumes have not met with widespread industry adoption. Consequently, Linux and Windows never detect an MBR volume as being larger than 2 TiB even if AWS shows its size to be larger.

GPT

GPT uses a 64-bit data structure to store block addresses. This means that each data block is mapped with one of 2^{64} possible integers. The maximum addressable size of a volume is given by the following formula:

$$2^{64} \times \text{Block size}$$

The block size for GPT volumes is commonly 4,096 bytes. Therefore:

$$\begin{aligned} & 2^{64} \times 4{,}096 \text{ bytes} \\ & \quad = 2^{64} \times 2^{12} \text{ bytes} \\ & \quad = 2^{70} \times 2^{6} \text{ bytes} \\ & \quad = 64 \text{ ZiB} \end{aligned}$$

Real-world computer systems don't support anything close to this theoretical maximum. Implemented file-system size is currently limited to 50 TiB for ext4 and 256 TiB for NTFS.

Data block sizes

Data storage on a modern hard drive is managed through *logical block addressing*, an abstraction layer that allows the operating system to read and write data in logical blocks without knowing much about the underlying hardware. The operating system relies on the storage device to map the blocks to its physical sectors, and reads and writes data to disk using data blocks that are a multiple of the sector size.

Amazon EBS advertises either 512-byte or 4,096-byte (4 KiB) physical sectors to the operating system, depending on the following factors:

1. The Amazon EC2 instance type
2. The operating system
3. The NVMe driver version

Amazon EBS advertises 4-KiB physical sectors only if all factors support it. If any one of these do not support 4-KiB physical sectors, Amazon EBS advertises 512-byte physical sectors.

Amazon EC2 instance type support

The following table shows the sector sizes that Amazon EBS advertises for the different Amazon EC2 instance types.

Instance type	Linux	Windows
All Xen-based instance types	Amazon EBS always advertises 512-byte physical sectors	
A1 | C5 | C5a | C5ad | C5d | C5n | C6g | C6gd | DL1 | D3 | D3en | G4ad | G4dn | G5 | G5g | I3 | I3en | Inf1 | M5 | M5a | M5ad | M5d | M5dn | M5n | M5zn | M6g | M6gd | P3dn | P4d | P4de | R5 | R5a | R5ad | R5d | R5dn | R5n | R6g | R6gd | T3 | T3a | T4g | U-12tb1 | U-18tb1 | U-24tb1 | U-3tb1 | U-6tb1 | U-9tb1 | X2gd | X2iezn | VT1 | Z1d	Amazon EBS always advertise s 512-byte physical sectors	Amazon EBS advertise s 512-byte or 4-KiB physical sectors [1]
All other Nitro-based instances	Amazon EBS advertises 512-byte or 4-KiB physical sectors [1]	

[1] Depends on the operating system support. See the following section.

Operating system support

The following table provides example operating systems and the corresponding physical sector sizes advertised by Amazon EBS. This is **not an exhaustive list**. We recommend that you verify the physical sector size advertised by Amazon EBS in your operating system.

Operating system	Advertised physical sector size
• Amazon Linux with kernel version 4.14 and earlier	512 byte

Operating system	Advertised physical sector size
• RHEL 7.9 and earlier • Ubuntu 20.04 and earlier • Windows 7/Windows Server 2008 and earlier	
• Amazon Linux with kernel version 5.3 and later • RHEL8.8 and later • Ubuntu 22.04 and later • Windows 8/Windows Server 2012 and later [1]	4 KiB

[1] For Windows workloads, ensure that you are using the latest version of the AWS NVMe drivers. Amazon EBS advertises 4-KiB physical sectors with AWS NVMe driver version 1.4.1 and later.

Non-default block sizes

The industry default size for logical data blocks is currently 4 KiB. Because certain workloads benefit from a smaller or larger block size, file systems support non-default block sizes that can be specified during formatting. Scenarios in which non-default block sizes should be used (such as optimizations) are outside the scope of this topic, but the choice of block size has consequences for the storage capacity of the volume. The following table shows theoretical storage capacity as a function of block size. However, note that the EBS-imposed limit on volume size (64 TiB for io2 Block Express) is currently equal to the maximum size enabled by 16-KiB data blocks.

Block size	Max volume size
4 KiB (default)	16 TiB
8 KiB	32 TiB
16 KiB	64 TiB
32 KiB	128 TiB

Block size	Max volume size
64 KiB (maximum)	256 TiB

Amazon EBS volumes and NVMe

Amazon EBS volumes are exposed as NVMe block devices on Amazon EC2 instances built on the AWS Nitro System. To fully utilize the performance and capabilities of Amazon EBS volumes exposed as NVMe block devices, the EC2 instance must have the AWS NVMe driver installed. All current generation AWS Windows and Linux AMIs come with the AWS NVMe driver installed by default.

If you use an AMI that does not have the AWS NVMe driver, you can manually install it. For more information, see AWS NVMe drivers in the *Amazon EC2 User Guide*.

Linux instances

The device names are `/dev/nvme0n1`, `/dev/nvme1n1`, and so on. The device names that you specify in a block device mapping are renamed using NVMe device names (`/dev/nvme[0-26]n1`). The block device driver can assign NVMe device names in a different order than you specified for the volumes in the block device mapping.

Windows instances

When you attach a volume to your instance, you include a device name for the volume. This device name is used by Amazon EC2. The block device driver for the instance assigns the actual volume name when mounting the volume, and the name assigned can be different than the name that Amazon EC2 uses.

Contents

Map Amazon EBS volumes to NVMe device names

EBS uses single-root I/O virtualization (SR-IOV) to provide volume attachments on Nitro-based instances using the NVMe specification. These devices rely on standard NVMe drivers on the operating system. These drivers typically discover attached devices during instance boot, and create device nodes based on the order in which the devices respond, not on how the devices are specified in the block device mapping.

Linux instances

In Linux, NVMe device names follow the pattern `/dev/nvme<x>n<y>`, where <x> is the enumeration order, and, for EBS, <y> is 1. Occasionally, devices can respond to discovery in a different order in subsequent instance starts, which causes the device name to change. Additionally, the device name assigned by the block device driver can be different from the name specified in the block device mapping.

We recommend that you use stable identifiers for your EBS volumes within your instance, such as one of the following:

- For Nitro-based instances, the block device mappings that are specified in the Amazon EC2 console when you are attaching an EBS volume or during `AttachVolume` or `RunInstances` API calls are captured in the vendor-specific data field of the NVMe controller identification. With Amazon Linux AMIs later than version 2017.09.01, we provide a `udev` rule that reads this data and creates a symbolic link to the block-device mapping.
- The EBS volume ID and the mount point are stable between instance state changes. The NVMe device name can change depending on the order in which the devices respond during instance boot. We recommend using the EBS volume ID and the mount point for consistent device identification.
- NVMe EBS volumes have the EBS volume ID set as the serial number in the device identification. Use the `lsblk -o +SERIAL` command to list the serial number.
- The NVMe device name format can vary depending on whether the EBS volume was attached during or after the instance launch. NVMe device names for volumes attached after instance launch include the `/dev/` prefix, while NVMe device names for volumes attached during instance launch do not include the `/dev/` prefix.
  - For Amazon Linux or FreeBSD AMI, use the `sudo ebsnvme-id /dev/nvme0n1 -u` command for a consistent NVMe device name.
  - For other distributions, use the `sudo nvme id-ctrl -V /dev/nvme0n1` command to determine the NVMe device name. You might need to include the `--vendor-specific` command option.

- When a device is formatted, a UUID is generated that persists for the life of the filesystem. A device label can be specified at the same time. For more information, see Make an Amazon EBS volume available for use and Boot from the wrong volume.

Amazon Linux AMIs

With Amazon Linux AMI 2017.09.01 or later (including Amazon Linux 2), you can run the **ebsnvme-id** command as follows to map the NVMe device name to a volume ID and device name:

The following example shows the command and output for a volume attached during instance launch. Note that the NVMe device name does not include the `/dev/` prefix.

```
[ec2-user ~]$ sudo /sbin/ebsnvme-id /dev/nvme0n1
Volume ID: vol-01324f611e2463981
sda
```

The following example shows the command and output for a volume attached after instance launch. Note that the NVMe device name includes the `/dev/` prefix.

```
[ec2-user ~]$ sudo /sbin/ebsnvme-id /dev/nvme1n1
Volume ID: vol-064784f1011136656
/dev/sdf
```

Amazon Linux also creates a symbolic link from the device name in the block device mapping (for example, `/dev/sdf`), to the NVMe device name.

FreeBSD AMIs

Starting with FreeBSD 12.2-RELEASE, you can run the **ebsnvme-id** command as shown above. Pass either the name of the NVMe device (for example, `nvme0`) or the disk device (for example, `nvd0` or `nda0`). FreeBSD also creates symbolic links to the disk devices (for example, `/dev/aws/disk/ebs/volume_id`).

Other Linux AMIs

With a kernel version of 4.2 or later, you can run the **nvme id-ctrl** command as follows to map an NVMe device to a volume ID. First, install the NVMe command line package, `nvme-cli`, using the package management tools for your Linux distribution. For download and installation instructions for other distributions, refer to the documentation specific to your distribution.

The following example gets the volume ID and NVMe device name for a volume that was attached during instance launch. Note that the NVMe device name does not include the /dev/ prefix. The device name is available through the NVMe controller vendor-specific extension (bytes 384:4095 of the controller identification):

```
[ec2-user ~]$ sudo nvme id-ctrl -V /dev/nvme0n1
NVME Identify Controller:
vid      : 0x1d0f
ssvid    : 0x1d0f
sn       : vol01234567890abcdef
mn       : Amazon Elastic Block Store
...
0000: 2f 64 65 76 2f 73 64 6a 20 20 20 20 20 20 20 20 "sda..."
```

The following example gets the volume ID and NVMe device name for a volume that was attached after instance launch. Note that the NVMe device name includes the /dev/ prefix.

```
[ec2-user ~]$ sudo nvme id-ctrl -V /dev/nvme1n1
NVME Identify Controller:
vid      : 0x1d0f
ssvid    : 0x1d0f
sn       : volabcdef01234567890
mn       : Amazon Elastic Block Store
...
0000: 2f 64 65 76 2f 73 64 6a 20 20 20 20 20 20 20 20 "/dev/sdf..."
```

The **lsblk** command lists available devices and their mount points (if applicable). This helps you determine the correct device name to use. In this example, /dev/nvme0n1p1 is mounted as the root device and /dev/nvme1n1 is attached but not mounted.

```
[ec2-user ~]$ lsblk
NAME           MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
nvme1n1        259:3   0  100G  0 disk
nvme0n1        259:0   0    8G  0 disk
  nvme0n1p1    259:1   0    8G  0 part /
  nvme0n1p128 259:2   0    1M  0 part
```

Windows instances

You can run the **ebsnvme-id** command to map the NVMe device disk number to an EBS volume ID and device name. By default, all EBS NVMe devices are enumerated. You can pass a disk number to enumerate information for a specific device. The ebsnvme-id tool is included in the latest AWS provided Windows Server AMIs located in C:\ProgramData\Amazon\Tools.

Starting with AWS NVMe driver package 1.5.0, the latest version of the ebsnvme-id tool is installed by the driver package. The latest version is only available in the driver package. The standalone download link for the ebsnvme-id tool will no longer receive updates. The last version available through the standalone link is 1.1.0, which can be downloaded using the link [ebsnvme-id.zip] and extracting the contents to your Amazon EC2 instance to get access to ebsnvme-id.exe.

```
PS C:\ProgramData\Amazon\Tools> ebsnvme-id.exe
Disk Number: 0
Volume ID: vol-0d6d7ee9f6e471a7f
Device Name: sda1

Disk Number: 1
Volume ID: vol-03a26248ff39b57cf
Device Name: xvdd

Disk Number: 2
Volume ID: vol-038bd1c629aa125e6
Device Name: xvde

Disk Number: 3
Volume ID: vol-034f9d29ec0b64c89
Device Name: xvdb

Disk Number: 4
Volume ID: vol-03e2dbe464b66f0a1
Device Name: xvdc
```

```
PS C:\ProgramData\Amazon\Tools> ebsnvme-id.exe 4
Disk Number: 4
Volume ID: vol-03e2dbe464b66f0a1
Device Name: xvdc
```

NVMe I/O operation timeout for Amazon EBS volumes

Most operating systems specify a timeout for I/O operations submitted to NVMe devices.

Linux instances

On Linux, EBS volumes attached to Nitro-based instances use the default NVMe driver provided by the operating system. Most operating systems specify a timeout for I/O operations submitted to NVMe devices. The default timeout is 30 seconds and can be changed using the `nvme_core.io_timeout` boot parameter. For most Linux kernels earlier than version 4.6, this parameter is `nvme.io_timeout`.

If I/O latency exceeds the value of this timeout parameter, the Linux NVMe driver fails the I/O and returns an error to the filesystem or application. Depending on the I/O operation, your filesystem or application can retry the error. In some cases, your filesystem might be remounted as read-only.

For an experience similar to EBS volumes attached to Xen instances, we recommend setting `nvme_core.io_timeout` to the highest value possible. For current kernels, the maximum is 4294967295, while for earlier kernels the maximum is 255. Depending on the version of Linux, the timeout might already be set to the supported maximum value. For example, the timeout is set to 4294967295 by default for Amazon Linux AMI 2017.09.01 and later.

You can verify the maximum value for your Linux distribution by writing a value higher than the suggested maximum to `/sys/module/nvme_core/parameters/io_timeout` and checking for the Numerical result out of range error when attempting to save the file.

Windows instances

On Windows, the default timeout is 60 seconds and the maximum is 255 seconds. You can modify the `TimeoutValue` disk class registry setting using the procedure described in [Registry Entries for SCSI Miniport Drivers](#).

NVMe Abort command for Amazon EBS volumes

The `Abort` command is an NVMe admin command that is issued to end a specific command that was previously submitted to the controller. This command is typically issued by the device driver to storage devices that have exceeded the I/O operation timeout threshold.

Amazon EC2 instance types that support the `Abort` command by default will end a specific command that was previously submitted to the controller when an `Abort` command is issued to attached Amazon EBS volumes. Amazon EC2 instances that do not support the `Abort` command take no action when an `Abort` command is issued to attached Amazon EBS volumes.

The `Abort` command is supported with:

- Amazon EBS devices with NVMe device version 1.4 or higher.

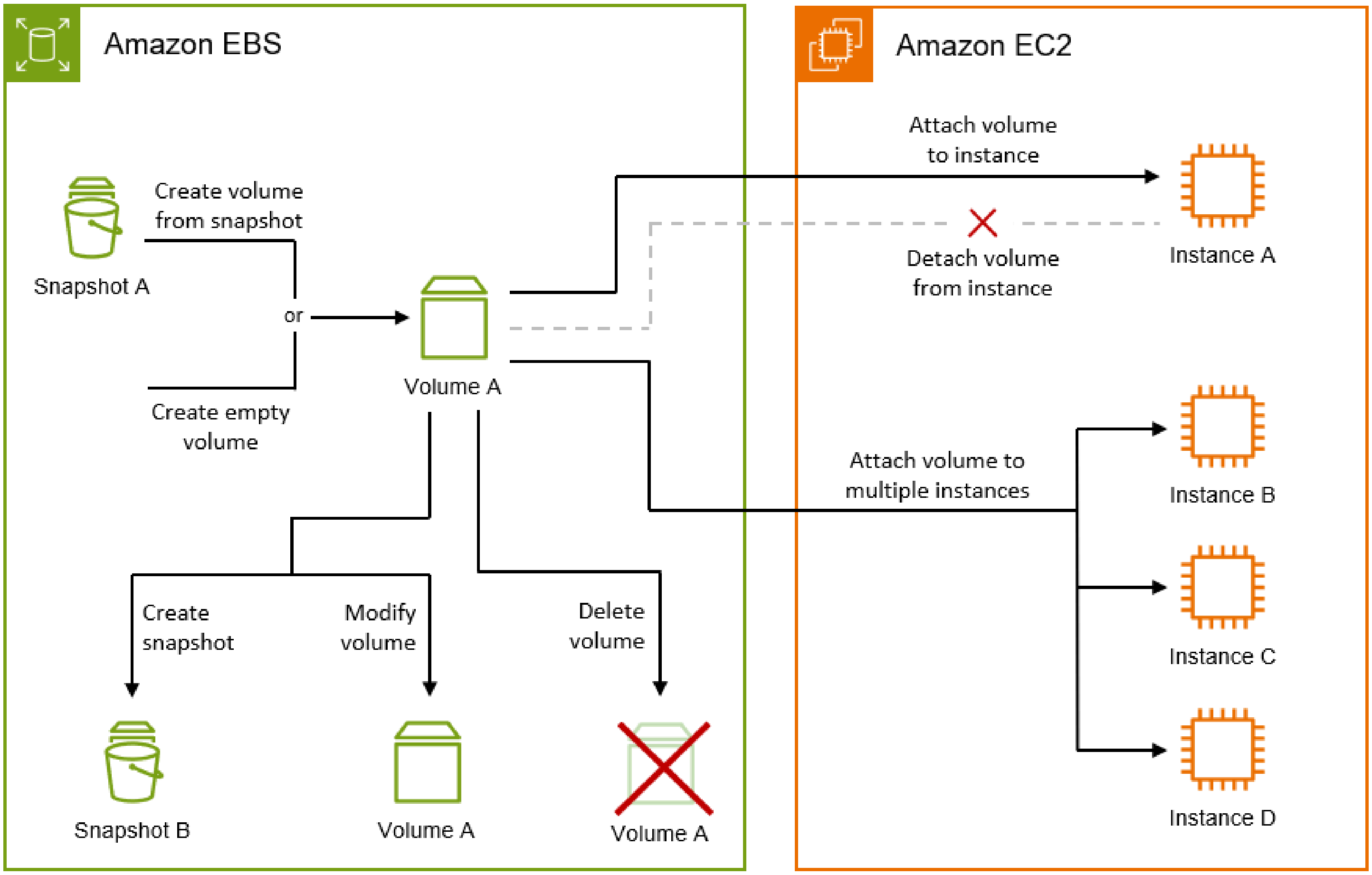
- All Amazon EC2 instances, **except** Xen-based instances types and the following Nitro-based instance types:
  - General purpose: A1 | M5 | M5a | M5ad | M5d | M5dn | M5n | M5zn | M6g | M6gd | Mac1 | Mac2 | T3 | T3a | T4g
  - Compute optimized: C5 | c5a | C5ad | C5d | C5n | C6g | C6gd
  - Memory optimized: R5 | R5a | R5ad | R5d | R5dn | R5n | R6g | R6gd | U-12tb1 | U-18tb1 | U-24tb1 | U-3tb1 | U-6tb1 | U-9tb1 | X2gd | X2iezn | Z1d
  - Storage optimized: D3 | D3en | I3en
  - Accelerated computing: DL1 | G4ad | G4dn | G5 | G5g | Inf1 | P3dn | P4d | P4de | VT1

For more information, see section **5.1 Abort command** of the NVM Express Base Specification.

Amazon EBS volume lifecycle

The lifecycle of an Amazon EBS volume starts with the creation process. You can create a volume from an Amazon EBS snapshot or you can create an empty volume. Before you can use your volume, you must attach it to one or more Amazon EC2 instances that are in the same Availability Zone as the volume. You can attach multiple volumes to an instance. If needed, you can detach a volume from one instance and then attach it to another instance. If your storage requirements change, you can modify the size or performance of the volume at any time. You can create point-in-time backups of your volumes by creating Amazon EBS snapshots. If you no longer need a volume, you can delete it to stop incurring the related storage costs.

The following image shows actions that you can perform on your volumes as part of the volume lifecycle. There are also tasks that you perform by connecting to the instance and running an operating system command. For example, formatting the volume, mounting the volume, managing partitions, and viewing the free disk space.



Tasks

- Create an Amazon EBS volume
- Copy an Amazon EBS volume
- Attach an Amazon EBS volume to an Amazon EC2 instance
- Attach an EBS volume to multiple EC2 instances using Multi-Attach
- Make an Amazon EBS volume available for use
- View information about an Amazon EBS volume
- Modify an Amazon EBS volume using Elastic Volumes operations
- Detach an Amazon EBS volume from an Amazon EC2 instance
- Delete an Amazon EBS volume

Create an Amazon EBS volume

You can create an Amazon EBS volume and then attach it to any EC2 instance in the same Availability Zone.

You can either **create an empty volume**, or you can **create a volume from an Amazon EBS snapshot**. If you create a volume from a snapshot, the volume begins as an exact replica of the volume that was used to create that snapshot.

Volume initialization

When you create a volume from a snapshot, the storage blocks from the snapshot must be downloaded from Amazon S3 and written to the volume before you can access them. This process is called volume initialization. During this time, the volume will experience increased I/O latency. Full volume performance is achieved only after all storage blocks have been downloaded and written to the volume.

The default volume initialization rate fluctuates throughout the initialization process, which could make completion times unpredictable.

To minimize the performance impacts associated with volume initialization, you can use an Amazon EBS Provisioned Rate for Volume Initialization (volume initialization rate) or fast snapshot restore. For more information, see Initialize Amazon EBS volumes.

Volume encryption

The encryption state of the volume depends on whether your account is enabled for encryption by default, and on the encryption state of the snapshot, if you choose to use one. The following table summarizes thepossible encryption outcomes.

Encryption by default	Snapshot used?	Volume encryption outcome	Note
Disabled	No	Optional encryption	If you enable encryption, you can specify the KMS key to use. If you enable encryption but do not specify a KMS key, the AWS managed key (`aws/ebs`) is used.

Encryption by default	Snapshot used?	Volume encryption outcome	Note
Disabled	Yes, unencrypted	Optional encryption	If you enable encryption, you can specify the KMS key to use. If you enable encryption but do not specify a KMS key, the AWS managed key (`aws/ebs`) is used.
Disabled	Yes, encrypted	Automatic encryption	You can specify the KMS key to use. If you do not specify a KMS key, the volume is encrypted using the same KMS key as the source snapshot.
Enabled	No	Automatic encryption	You can specify the KMS key to use. If you do not specify a KMS key, the key specified for encryption by default is used.
Enabled	Yes, unencrypted	Automatic encryption	You can specify the KMS key to use. If you do not specify a KMS key, the key specified for encryption by default is used.
Enabled	Yes, encrypted	Automatic encryption	You can specify the KMS key to use. If you do not specify a KMS key, the volume is encrypted using the same key as the source snapshot (console), or the key specified for encryption by default (CLI/API).

Additional considerations

- Volumes must be attached to instances in the same Availability Zone.
- Volumes are ready for use only after they enter the `available` state.

- When you create a volume using the console, gp3 is the default volume type. For the command line tools, API, and SDK, gp2 is the default volume type.
- To use a volume with an instance running on an outpost, you must create the volume on the same outpost as the instance.
- If you create a volume for use with a Windows instance, and it's larger than 2048 GiB, ensure that you configure the volume to use GPT partition tables. For more information, see Amazon EBS volume constraints and Windows support for disks larger than 2 TB..
- Volumes are also created indirectly by launching an Amazon EC2 instance. Either the AMI used to launch the instance, or the instance launch request itself could include block device mappings for Amazon EBS volumes. For more information, see Block device mappings.

Console

To create a volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes** and then choose **Create volume**.
3. (*Outpost customers only*) For **Outpost ARN**, enter the ARN of the AWS Outpost on which to create the volume.
4. For **Volume type**, choose the type of volume to create. For more information about the available volume types, see Amazon EBS volume types.
5. For **Size**, enter the size of the volume, in GiB. For more information, see Amazon EBS volume constraints.
6. (*For `io1`, `io2`, and gp3 only*) For **IOPS**, enter the maximum number of input/output operations per second (IOPS) that the volume should provide.
7. (*For gp3 only*) For **Throughput**, enter the throughput that the volume should provide, in MiB/s.
8. For **Availability Zone**, choose the Availability Zone in which to create the volume.
9. For **Snapshot ID**, do one of the following:
   - To create an empty volume, keep the default value (**Don't create volume from a snapshot**).
   - To create the volume from a snapshot, select the snapshot to use.

10. If you have selected a snapshot, for **Volume initialization rate**, you can optionally specify the volume initialization rate, in MiB/s, at which the snapshot blocks are to be downloaded from Amazon S3 to the volume after creation. For more information, see Use an Amazon EBS Provisioned Rate for Volume Initialization. To use the default initialization rate or fast snapshot restore (if it is enabled for the selected snapshot), don't specify a rate.
11. (*`io1` and `io2` only*) To enable the volume for Amazon EBS Multi-Attach, select **Enable Multi-Attach**. For more information, see Attach an EBS volume to multiple EC2 instances using Multi-Attach.
12. Set the encryption status for the volume.
    - If your account is enabled for encryption by default, encryption is automatic and can't be disabled.
    - If you selected an encrypted snapshot, encryption is automatic and can't be disabled.
    - If your account is not enabled for encryption by default, and you select an unencrypted snapshot or do not select a snapshot, encryption is optional.
13. (*Optional*) To assign custom tags to the volume, in the **Tags** section, choose **Add tag**, and then enter a tag key and value pair.
14. Choose **Create volume**.
15. To use the volume, wait for it to reach the `available` state and then attach it to an Amazon EC2 instance in the same Availability Zone. For more information, see Attach an Amazon EBS volume to an Amazon EC2 instance.

AWS CLI

To create a volume

Use the create-volume command. The following example creates an empty gp3 volume with a size of 100 GiB in the specified Availability Zone.

```
aws ec2 create-volume \
    --volume-type gp3 \
    --size 100 \
    --availability-zone us-east-1a
```

PowerShell

To create a volume

Use the New-EC2Volume cmdlet. The following example creates an empty gp3 volume with a size of 100 GiB in the specified Availability Zone.

```
New-EC2Volume `
    -VolumeType gp3 `
    -Size 100 `
    -AvailabilityZone us-east-1a
```

Copy an Amazon EBS volume

You can create an instant point-in-time copy of an Amazon EBS volume within the same Availability Zone. A volume copy begins as a crash-consistent, point in time copy of the source volume. It includes all the data blocks written to the source volume at the time the volume copy initialization begins. The volume copy gets its own unique volume ID. Volume copies are created immediately and can be attached to an Amazon EC2 instance once it reaches the `available` state. Using volume copies, you can quickly copy your production data for test and development environments.

Initialization

Volume copies are initialized after creation. During initialization, the data blocks are copied from the source volume and written to the volume copy in the background. The volume remains in the `initializing` state until initialization completes.

Performance during initialization

Copy operations do not affect the performance of the source volume. You can continue using the source volume normally during the copy process. Copied volumes can be accessed instantly without waiting for the data to be copied from the source volume. Volume copies provide instant access to data with single-digit millisecond latency, however, actual latency might vary depending on the volume type. During initialization, the volume copy delivers **baseline performance** equal to the lowest of the following three values:

- 3,000 IOPS and 125 MiB/s
- The provisioned performance for the **source volume**
- The provisioned performance for the **volume copy**

The volume copy can exceed the baseline performance when the following criteria are met:

1. Both the source volume and volume copy are provisioned with more than 3,000 IOPS and 125 MiB/s.
2. The source volume has unutilized performance capacity (driven performance is less than provisioned performance)

For example, if the source volume is provisioned with 10,000 IOPS and your workload is currently driving only 5,000 IOPS, and the volume copy is provisioned with 10,000 IOPS, the volume copy can achieve performance higher than the 3,000 IOPS baseline performance during initialization by using the source volume's unutilized 5,000 IOPS.

Initialization duration

The time it takes to initialize a volume copy depends on the size of the block data written to the source volume at the time of creating the volume copy. Volume copies are initialized on a best-effort basis, with the following general guidelines. For the first 1 TiB of data blocks, volume initialization takes up to 6 hours. For each subsequent 1 TiB of data blocks up to 16 TiB, initialization takes 1.2 hours per TiB. For written data larger than 16 TiB, initialization takes 24 hours.

Monitor initialization progress

You can monitor the initialization progress using the describe-volume-status AWS CLI command or Amazon EventBridge. For more information, see Monitor the status of Amazon EBS volume initialization.

Encryption

Copies of encrypted volumes are automatically encrypted with the same KMS key as the source volume. You can't copy unencrypted volumes.

Considerations

- You can create copies from encrypted source volumes only. You can't create copies from unencrypted source volumes.
- You can create only one volume copy from a source volume at a time. You can create subsequent copies of the same source volume only once the previous volume copy has been fully initialized.
- You can have a maximum of 5 in-progress volume copies per Region. If you exceed this quota, you get the `CopyVolumesLimitExceeded` error. You can request a quota increase if needed.
- The volume copy must be created in the same Availability Zone as the source volume.

- The size of the volume copy must be equal to or greater than the size of the source volume.
- You can't copy a volume copy while it is being created or initialized.
- To create a volume copy, the source volume must be in the `available` or `in-use` state, and volume modifications must be in the `completed` or `optimizing` state.
- Volume copies are subject to the same account and Regional storage and IOPS quotas as regular Amazon EBS volumes. For more information, see Amazon EBS quotas.
- If you delete the source volume while the copy operation is in progress, the copy operation still completes.
- Tags assigned to the source volume are not assigned to the volume copy.
- You can't create copies from volumes on Outposts or in Wavelength Zones.

Pricing

When you initiate volume copy operation, you are charged a one-time fee per GiB of data blocks written to the volume copy. After the volume copy is created, it is charged the same way as any other Amazon EBS volume in your account. For more information, see Amazon EBS pricing.

Copy a volume

Use one of the following methods to copy an Amazon EBS volume.

Console

To copy a volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume to copy and choose **Actions**, **Copy volume**.
4. For **Volume type**, choose the volume type for the copy. The default volume type is **gp3**.
5. For **Size**, enter the size for the volume copy, in GiBs. The size must be equal to or greater than the size of the source volume.
6. (*io1, io2, and gp3 only*) For **IOPS**, enter the maximum number of input/output operations per second (IOPS) for the volume copy.
7. (*gp3 only*) For **Throughput**, enter the throughput for the volume copy, in MiB/s.
8. (*io1 and io2 only*) To enable the volume copy for Amazon EBS Multi-Attach, select **Enable Multi-Attach**.

9. (*Optional*) To assign custom tags to the volume copy, in the **Tags** section, choose **Add tag**, and then enter a tag key and value pair.
10. Choose **Copy volume**.
11. The copied volume enters the `creating` state and then transitions to `available` shortly after. You can then attach it to an Amazon EC2 instance in the same Availability Zone.

AWS CLI

To copy a volume

Use the copy-volumes command.

The following example creates a volume copy of `vol-01234567890abcdef` with the gp3 volume type, a size of `100` GiB, and throughput of `250` MiB/s.

```
aws ec2 copy-volumes \
--source-volume-id vol-01234567890abcdef \
--volume-type gp3 \
--size 100 \
--throughput 250
```

PowerShell

To copy a volume

Use the Copy-EC2Volume cmdlet.

The following example creates a volume copy of `vol-01234567890abcdef` with the gp3 volume type, a size of `100` GiB, and throughput of `250` MiB/s.

```
Copy-EC2Volume `
-SourceVolumeId vol-01234567890abcdef `
-VolumeType gp3 `
-Size 100 `
-Throughput 250
```

Attach an Amazon EBS volume to an Amazon EC2 instance

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

For information about adding EBS volumes to your instance at launch, see instance block device mapping.

Considerations

- The maximum number of Amazon EBS volumes that you can attach to an instance depends on the instance type. If you exceed the volume attachment limit for an instance type, the attachment request fails with the `AttachmentLimitExceeded` error. For more information, see Instance volume limits.
- You can attach volumes to instances that are in the same Availability Zone only.
- Multi-Attach enabled volumes can be attached to up to 16 instances. For more information, see Attach an EBS volume to multiple EC2 instances using Multi-Attach.
- If the volume has an AWS Marketplace product code:
  - You can attach it to a stopped instance only.
  - You must be subscribed to the AWS Marketplace code that is on the volume.
  - The instance's configuration, such as its type and operating system, must support that specific AWS Marketplace code. For example, you cannot take a volume from a Windows instance and attach it to a Linux instance.
  - AWS Marketplace codes are copied from the volume to the instance.
- This device name you specify is used by Amazon EC2. The block device driver can mount the device with a device name that is different from the one you specify. For more information, see Device names for volumes on Amazon EC2 instances.
- In some cases, a volume other than the volume attached to `/dev/xvda` or `/dev/sda` can become the root volume for the instance. This can happen if you attach the root volume of another instance, or a volume created from the snapshot of a root volume, to an instance with an existing root volume. For more information, see Boot from the wrong volume.
- Some instance types support more than one EBS card. You can select the EBS card for the volume to be attached to by specifying the EBS card index. For instances support multiple EBS cards, see EBS cards.
  - Your root volume must be attached to EBS card index `0`.
  - For the instances that support multiple EBS cards, If you do not specify the EBS card index, your volume will be attached to EBS card index `0`.
  - When configuring your EC2 instances for high-performance workloads, it is essential to balance EBS volumes across EBS cards based on performance requirements, to avoid running into performance limits on any of the EBS cards.

- The volume attachment limit for an instance type is spread equally across each EBS card. For example, on an EC2 instance that supports 128 volume attachments with 2 EBS cards, each EBS card can support up to 64 volume attachments. If you exceed the EBS card attachment limit, the request fails with the `CardAttachmentLimitExceeded` error.

Console

To attach an EBS volume to an instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume to attach and choose **Actions**, **Attach volume**.
4. For **Instance**, enter the ID of the instance or select the instance from the list of options.
5. For **Device name**, do one of the following:
   - For a root volume, select the required device name from the **Reserved for root volume** section of the list. Typically `/dev/sda1` or `/dev/xvda` for Linux instances depending on the AMI, or `/dev/sda1` for Windows instances.
   - For data volumes, select an available device name from the **Recommended for data volumes** section of the list.
   - To use a custom device name, select **Specify a custom device name** and then enter the device name to use.
6. Choose **Attach volume**.
7. Connect to the instance and mount the volume. For more information, see [Make an Amazon EBS volume available for use](#).

AWS CLI

To attach an EBS volume to an instance

Use the [attach-volume](#) command. The following example attaches the specified volume to the specified instance using the specified device name.

```
aws ec2 attach-volume \
    --volume-id vol-01234567890abcdef \
    --instance-id i-1234567890abcdef0 \
```

```
    --device /dev/sdf
```

PowerShell

To attach an EBS volume to an instance

Use the Add-EC2Volume cmdlet. The following example attaches the specified volume to the specified instance using the specified device name.

```
Add-EC2Volume `
    -VolumeId vol-01234567890abcdef `
    -InstanceId i-1234567890abcdef0 `
    -Device /dev/sdf
```

Attach an EBS volume to multiple EC2 instances using Multi-Attach

Amazon EBS Multi-Attach enables you to attach a single Provisioned IOPS SSD (`io1` or `io2`) volume to multiple instances that are in the same Availability Zone. You can attach multiple Multi-Attach enabled volumes to an instance or set of instances. Each instance to which the volume is attached has full read and write permission to the shared volume. Multi-Attach makes it easier for you to achieve higher application availability in applications that manage concurrent write operations.

Pricing and billing

There are no additional charges for using Amazon EBS Multi-Attach. You are billed the standard charges that apply to Provisioned IOPS SSD (`io1` and `io2`) volumes. For more information, see Amazon EBS pricing.

Contents

Considerations and limitations

- Multi-Attach enabled volumes can be attached to up to 16 instances built on the Nitro System that are in the same Availability Zone.
- **Linux instances** support Multi-Attach enabled `io1` and `io2` volumes. **Windows instances** support Multi-Attach enabled `io2` volumes only.
- The maximum number of Amazon EBS volumes that you can attach to an instance depends on the instance type and instance size. For more information, see instance volume limits.
- Multi-Attach is supported exclusively on Provisioned IOPS SSD (`io1` and `io2`) volumes.
- Multi-Attach for `io1` volumes is available in the following Regions only: US East (N. Virginia), US West (Oregon), and Asia Pacific (Seoul).

  Multi-Attach for `io2` is available in all Regions that support `io2`.

  > **Note**
  >
  > For better performance, consistency, and durability at a lower cost, we recommend that you use `io2` volumes.

- `io1` volumes with Multi-Attach enabled are not supported with instances built on the Nitro System that support the Scalable Reliable Datagram (SRD) networking protocol only. To use Multi-Attach with these instance types, you must use `io2`.
- Standard file systems, such as XFS and EXT4, are not designed to be accessed simultaneously by multiple servers, such as EC2 instances. You should use a clustered file system to ensure data resiliency and reliability for your production workloads.
- Multi-Attach enabled `io2` volumes support I/O fencing. I/O fencing protocols control write access in a shared storage environment to maintain data consistency. Your applications must provide write ordering for the attached instances to maintain data consistency. For more information, see Use NVMe reservations with Multi-Attach enabled Amazon EBS volumes.

  Multi-Attach enabled `io1` volumes do not support I/O fencing.
- Multi-Attach enabled volumes can't be created as boot volumes.
- Multi-Attach enabled volumes can be attached to one block device mapping per instance.
- Multi-Attach can't be enabled during instance launch using either the Amazon EC2 console or RunInstances API.

- Multi-Attach enabled volumes that have an issue at the Amazon EBS infrastructure layer are unavailable to all attached instances. Issues at the Amazon EC2 or networking layer might impact only some attached instances.
- The following table shows volume modification support for Multi-Attach enabled `io1` and `io2` volumes after creation.

	`io2` volumes	**`io1` volumes**
Modify volume type	✗	✗
Modify volume size	✓	✗
Modify provisioned IOPS	✓	✗
Enable Multi-Attach	✓ *	✗
Disable Multi-Attach	✓ *	✗

  * You can't enable or disable Multi-Attach while the volume is attached to an instance.
- Multi-Attach enabled volumes are deleted on instance termination if the last attached instance is terminated and if that instance is configured to delete the volume on termination. If the volume is attached to multiple instances that have different delete on termination settings in their volume block device mappings, the last attached instance's block device mapping setting determines the delete on termination behavior.

  To ensure predictable delete on termination behavior, enable or disable delete on termination for all of the instances to which the volume is attached. For more information, see [Preserve data when an instance is terminated](#).
- You can monitor a Multi-Attach enabled volume using the CloudWatch Metrics for Amazon EBS volumes. Data is aggregated across all of the attached instances. You can't monitor metrics

for individual attached instances. For more information, see Amazon CloudWatch metrics for Amazon EBS.

Performance for Multi-Attach Amazon EBS volumes

Each attached instance is able to drive its maximum IOPS performance up to the volume's maximum provisioned performance. However, the aggregate performance of all of the attached instances can't exceed the volume's maximum provisioned performance. If the attached instances' demand for IOPS is higher than the volume's Provisioned IOPS, the volume will not exceed its provisioned performance.

For example, say you create an `io2` Multi-Attach enabled volume with `80,000` provisioned IOPS and you attach it to an `m7g.large` instance that supports up to `40,000` IOPS, and an `r7g.12xlarge` instance that supports up to `60,000` IOPS. Each instance can drive its maximum IOPS as it is less than the volume's Provisioned IOPS of `80,000`. However, if both instances drive I/O to the volume simultaneously, their combined IOPS can't exceed the volume's provisioned performance of `80,000` IOPS.

To achieve consistent performance, it is best practice to balance I/O driven from attached instances across the sectors of a Multi-Attach enabled volume.

For more information about IOPS performance for the Amazon EC2 instance types, see Amazon EBS optimized instance types in the *Amazon EC2 User Guide*.

Enable Multi-Attach for an Amazon EBS volume

Multi-Attach enabled volumes can be managed in much the same way that you would manage any other Amazon EBS volume. However, in order to use the Multi-Attach functionality, you must enable it for the volume.

When you create a new volume, Multi-Attach is disabled by default. You can enable Multi-Attach when you create a volume.

You can also enable Multi-Attach for `io2` volumes after creation, but only if they are not attached to any instances. You can't enable Multi-Attach for `io1` volumes after creation.

After you enable Multi-Attach for a volume, you can attach the volume to an instance in the same way that you attach any other EBS volume. For more information, see Attach an Amazon EBS volume to an Amazon EC2 instance.

Console

To enable Multi-Attach during volume creation

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Choose **Create volume**.
4. For **Volume type**, choose **Provisioned IOPS SSD (io1)** or **Provisioned IOPS SSD (io2)**.
5. For **Size** and **IOPS**, choose the required volume size and the number of IOPS to provision.
6. For **Availability Zone**, choose the same Availability Zone that the instances are in.
7. For **Amazon EBS Multi-Attach**, choose **Enable Multi-Attach**.
8. (Optional) For **Snapshot ID**, choose the snapshot from which to create the volume.
9. Set the encryption status for the volume.

   If the selected snapshot is encrypted, or if your account is enabled for encryption by default, then encryption is automatically enabled and you can't disable it. You can choose the KMS key to use to encrypt the volume.

   If the selected snapshot is unencrypted and your account is not enabled for encryption by default, encryption is optional. To encrypt the volume, for **Encryption**, choose **Encrypt this volume** and then select the KMS key to use to encrypt the volume.

   You can attach encrypted volumes only to instances that support Amazon EBS encryption. For more information, see Amazon EBS encryption.
10. (Optional) To assign custom tags to the volume, in the **Tags** section, choose **Add tag**, and then enter a tag key and value pair.
11. Choose **Create volume**.

To enable Multi-Attach after creation

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume and choose **Actions**, **Modify volume**.
4. For **Amazon EBS Multi-Attach**, choose **Enable Multi-Attach**.
5. Choose **Modify**.

AWS CLI

To enable Multi-Attach during volume creation

Use the create-volume command with the `--multi-attach-enabled` option.

```
aws ec2 create-volume \
    --volume-type io2 \
    --multi-attach-enabled \
    --size 100 \
    --iops 2000 \
    --region us-west-2 \
    --availability-zone us-west-2b
```

To enable Multi-Attach after creation

Use the modify-volume command with the `--multi-attach-enabled` option.

```
aws ec2 modify-volume \
    --volume-id vol-01234567890abcdef \
    --multi-attach-enabled
```

PowerShell

To enable Multi-Attach during volume creation

Use the New-EC2Volume cmdlet with the `-MultiAttachEnabled` parameter.

```
New-EC2Volume `
    -VolumeType io2 `
    -MultiAttachEnabled $true `
    -Size 100 `
    -Iops 2000 `
    -Region us-west-2 `
    -AvailabilityZone us-west-2b
```

To enable Multi-Attach after creation

Use the Edit-EC2Volume cmdlet with the `-MultiAttachEnabled` parameter.

```
Edit-EC2Volume `
    -VolumeId vol-01234567890abcdef `
    -MultiAttachEnabled $true
```

Disable Multi-Attach for an Amazon EBS volume

You can disable Multi-Attach for an `io2` volume only if it is attached to no more than one instance.

You can't disable Multi-Attach for `io1` volumes after creation.

Console

To disable Multi-Attach after creation

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume and choose **Actions**, **Modify volume**.
4. For **Amazon EBS Multi-Attach**, clear **Enable Multi-Attach**.
5. Choose **Modify**.

AWS CLI

To disable Multi-Attach after creation

Use the modify-volume command with the `-no-multi-attach-enabled` option.

```
aws ec2 modify-volume \
    --volume-id vol-01234567890abcdef \
    --no-multi-attach-enabled
```

PowerShell

To disable Multi-Attach after creation

Use the Edit-EC2Volume cmdlet with the `-MultiAttachEnabled` parameter.

```
Edit-EC2Volume `
    -VolumeId vol-01234567890abcdef `
    -MultiAttachEnabled $false
```

Use NVMe reservations with Multi-Attach enabled Amazon EBS volumes

Multi-Attach enabled `io2` volumes support NVMe reservations, which is a set of industry-standard storage fencing protocols. These protocols enable you to create and manage reservations that control and coordinate access from multiple instances to a shared volume. Reservations are used by shared storage applications to ensure data consistency.

Contents

Requirements

NVMe reservations is supported with Multi-Attach enabled `io2` volumes only. Multi-Attach enabled volumes can be attached only to instances built on the Nitro system.

NVMe reservations is supported with the following operating systems:

- SUSE Linux Enterprise 12 SP3 and later
- RHEL 8.3 and later
- Amazon Linux 2 and later
- Windows Server 2016 and later

> **Note**
>
> For supported Windows Server AMIs dated 2023.09.13 and later, the required NVMe drivers are included. For earlier AMIs, you must update to NVMe driver version 1.5.0 or later. For more information, see AWS NVMe drivers.

If you're using EC2Launch v2 to initialize your disks, you must upgrade to version **2.0.1521** or later. For more information, see Use the EC2Launch v2 agent.

Enabling support for NVMe reservations

Support for NVMe reservations is enabled by default for all Multi-Attach enabled `io2` volumes created after **September 18, 2023**.

To enable support for NVMe reservations for existing `io2` volumes created before September 18, 2023, you must detach all instances from the volume and then reattach the required instances. All attachments made after detaching all of the instances will have NVMe reservations enabled.

Supported NVMe Reservation commands

Amazon EBS supports the following NVMe Reservation commands:

Reservation Register

Registers, unregisters, or replaces a reservation key. A registration key is used to identify and authenticate an instance. Registering a reservation key with a volume creates an association between the instance and the volume. You must register the instance with the volume before that instance can acquire a reservation.

Reservation Acquire

Acquires a reservation on a volume, preempts a reservation held on a namespace, and aborts a reservation held on a volume. The following reservation types can be acquired:

- Write Exclusive Reservation
- Exclusive Access Reservation
- Write Exclusive - Registrants Only Reservation
- Exclusive Access - Registrants Only Reservation
- Write Exclusive - All Registrants Reservation
- Exclusive Access - All Registrants Reservation

Reservation Release

Releases or clears a reservation held on a volume.

Reservation Report

Describes the registration and reservation status of a volume.

Pricing

There are no additional costs for enabling and using Multi-Attach.

Make an Amazon EBS volume available for use

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

You can take snapshots of your EBS volume for backup purposes or to use as a baseline when you create another volume. For more information, see Amazon EBS snapshots.

If the EBS volume you are preparing for use is greater than 2 TiB, you must use a GPT partitioning scheme to access the entire volume. For more information, see Amazon EBS volume constraints.

Linux instances

Format and mount an attached volume

Suppose that you have an EC2 instance with an EBS volume for the root device, `/dev/xvda`, and that you have just attached an empty EBS volume to the instance using `/dev/sdf`. Use the following procedure to make the newly attached volume available for use.

To format and mount an EBS volume on Linux

1. Connect to your instance using SSH. For more information, see Connect to your Linux instance.
2. The device could be attached to the instance with a different device name than you specified in the block device mapping. For more information, see device names on Linux instances. Use the **lsblk** command to view your available disk devices and their mount points (if applicable) to help you determine the correct device name to use. The output of **lsblk** removes the `/dev/` prefix from full device paths.

   The following is example output for an instance built on the Nitro System, which exposes EBS volumes as NVMe block devices. The root device is `/dev/nvme0n1`, which has two partitions named `nvme0n1p1` and `nvme0n1p128`. The attached volume is `/dev/nvme1n1`, which has no partitions and is not yet mounted.

   ```
   [ec2-user ~]$ lsblk
   NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
   nvme1n1       259:0    0  10G  0 disk
   nvme0n1       259:1    0   8G  0 disk
   -nvme0n1p1    259:2    0   8G  0 part /
   -nvme0n1p128  259:3    0   1M  0 part
   ```

The following is example output for a T2 instance. The root device is /dev/xvda, which has one partition named xvda1. The attached volume is /dev/xvdf, which has no partitions and is not yet mounted.

```
[ec2-user ~]$ lsblk
NAME    MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
xvda    202:0    0    8G  0 disk
-xvda1  202:1    0    8G  0 part /
xvdf    202:80   0   10G  0 disk
```

3. Determine whether there is a file system on the volume. New volumes are raw block devices, and you must create a file system on them before you can mount and use them. Volumes that were created from snapshots likely have a file system on them already; if you create a new file system on top of an existing file system, the operation overwrites your data.

   Use one or both of the following methods to determine whether there is a file system on the volume:

   - Use the **file -s** command to get information about a specific device, such as its file system type. If the output shows simply data, as in the following example output, there is no file system on the device

     ```
     [ec2-user ~]$ sudo file -s /dev/xvdf
     /dev/xvdf: data
     ```

     If the device has a file system, the command shows information about the file system type. For example, the following output shows a root device with the XFS file system.

     ```
     [ec2-user ~]$ sudo file -s /dev/xvda1
     /dev/xvda1: SGI XFS filesystem data (blksz 4096, inosz 512, v2 dirs)
     ```

   - Use the **lsblk -f** command to get information about all of the devices attached to the instance.

     ```
     [ec2-user ~]$ sudo lsblk -f
     ```

     For example, the following output shows that there are three devices attached to the instances—nvme1n1, nvme0n1, and nvme2n1. The first column lists the devices and their partitions. The FSTYPE column shows the file system type for each device. If the column is empty for a specific device, it means that the device does not have a file system. In this case, device nvme1n1 and partition nvme0n1p1 on device nvme0n1 are both formatted using the XFS file system, while device nvme2n1 and partition nvme0n1p128 on device nvme0n1 do not have file systems.

```
NAME   FSTYPE LABEL UUID       MOUNTPOINT
nvme1n1         xfs   7f939f28-6dcc-4315-8c42-6806080b94dd
nvme0n1
##nvme0n1p1 xfs     / 90e29211-2de8-4967-b0fb-16f51a6e464c         /
##nvme0n1p128
nvme2n1
```

If the output from these commands show that there is no file system on the device, you must create one.

4. (Conditional) If you discovered that there is a file system on the device in the previous step, skip this step. If you have an empty volume, use the **mkfs -t** command to create a file system on the volume.

> **Warning**
>
> Do not use this command if you're mounting a volume that already has data on it (for example, a volume that was created from a snapshot). Otherwise, you'll format the volume and delete the existing data.

```
[ec2-user ~]$ sudo mkfs -t xfs /dev/xvdf
```

If you get an error that mkfs.xfs is not found, use the following command to install the XFS tools and then repeat the previous command:

```
[ec2-user ~]$ sudo yum install xfsprogs
```

5. Use the **mkdir** command to create a mount point directory for the volume. The mount point is where the volume is located in the file system tree and where you read and write files to after you mount the volume. The following example creates a directory named /data.

```
[ec2-user ~]$ sudo mkdir /data
```

6. Mount the volume or partition at the mount point directory you created in the previous step.

   If the volume has no partitions, use the following command and specify the device name to mount the entire volume.

   ```
   [ec2-user ~]$ sudo mount /dev/xvdf /data
   ```

   If the volume has partitions, use the following command and specify the partition name to mount a partition.

   ```
   [ec2-user ~]$ sudo mount /dev/xvdf1 /data
   ```

7. Review the file permissions of your new volume mount to make sure that your users and applications can write to the volume. For more information about file permissions, see File security at *The Linux Documentation Project*.
8. The mount point is not automatically preserved after rebooting your instance. To automatically mount this EBS volume after reboot, follow the next procedure.

Automatically mount an attached volume after reboot

To mount an attached EBS volume on every system reboot, add an entry for the device to the /etc/fstab file.

You can use the device name, such as /dev/xvdf, in /etc/fstab, but we recommend using the device's 128-bit universally unique identifier (UUID) instead. Device names can change, but the UUID persists throughout the life of the partition. By using the UUID, you reduce the chances that the system becomes unbootable after a hardware reconfiguration. For more information, see Map Amazon EBS volumes to NVMe device names.

To mount an attached volume automatically after reboot

1. (Optional) Create a backup of your /etc/fstab file that you can use if you accidentally destroy or delete this file while editing it.

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

2. Use the **blkid** command to find the UUID of the device. Make a note of the UUID of the device that you want to mount after reboot. You'll need it in the following step.

   For example, the following command shows that there are two devices mounted to the instance, and it shows the UUIDs for both devices.

```
[ec2-user ~]$ sudo blkid
/dev/xvda1: LABEL="/" UUID="ca774df7-756d-4261-a3f1-76038323e572" TYPE="xfs"
 PARTLABEL="Linux" PARTUUID="02dcd367-e87c-4f2e-9a72-a3cf8f299c10"
/dev/xvdf: UUID="aebf131c-6957-451e-8d34-ec978d9581ae" TYPE="xfs"
```

   For Ubuntu 18.04 use the lsblk command.

```
[ec2-user ~]$ sudo lsblk -o +UUID
```

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

```
[ec2-user ~]$ sudo vim /etc/fstab
```

4. Add the following entry to `/etc/fstab` to mount the device at the specified mount point. The fields are the UUID value returned by **blkid** (or **lsblk** for Ubuntu 18.04), the mount point, the file system, and the recommended file system mount options. For more information about the required fields, run `man fstab` to open the **fstab** manual.

   In the following example, we mount the device with UUID `aebf131c-6957-451e-8d34-ec978d9581ae` to mount point `/data` and we use the `xfs` file system. We also use the `defaults` and `nofail` flags. We specify `0` to prevent the file system from being dumped, and we specify 2 to indicate that it is a non-root device.

```
UUID=aebf131c-6957-451e-8d34-ec978d9581ae  /data  xfs  defaults,nofail  0  2
```

   Note

   If you ever boot your instance without this volume attached (for example, after moving the volume to another instance), the `nofail` mount option enables the instance to boot even if there are errors mounting the volume. Debian derivatives, including Ubuntu versions earlier than 16.04, must also add the `nobootwait` mount option.

5. To verify that your entry works, run the following commands to unmount the device and then mount all file systems in `/etc/fstab`. If there are no errors, the `/etc/fstab` file is OK and your file system will mount automatically after it is rebooted.

```
[ec2-user ~]$ sudo umount /data
[ec2-user ~]$ sudo mount -a
```

   If you receive an error message, address the errors in the file.

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

   If you are unsure how to correct errors in `/etc/fstab` and you created a backup file in the first step of this procedure, you can restore from your backup file using the following command.

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

Windows instances

Use one of the following methods to make a volume available on a Windows instance.

PowerShell

To make all EBS volumes with raw partitions available to use with Windows PowerShell

1. Log in to your Windows instance using Remote Desktop. For more information, see [Connect to your Windows instance](#).
2. On the taskbar, open the Start menu, and choose **Windows PowerShell**.
3. Use the provided series of Windows PowerShell commands within the opened PowerShell prompt. The script performs the following actions by default:
   1. Stops the ShellHWDetection service.
   2. Enumerates disks where the partition style is raw.

3. Creates a new partition that spans the maximum size the disk and partition type will support.
4. Assigns an available drive letter.
5. Formats the file system as NTFS with the specified file system label.
6. Starts the ShellHWDetection service again.

```
Stop-Service -Name ShellHWDetection
Get-Disk | Where PartitionStyle -eq 'raw' | Initialize-Disk -PartitionStyle MBR
 -PassThru | New-Partition -AssignDriveLetter -UseMaximumSize | Format-Volume -
FileSystem NTFS -NewFileSystemLabel "Volume Label" -Confirm:$false
Start-Service -Name ShellHWDetection
```

DiskPart command line tool

To make an EBS volume available to use with the DiskPart command line tool

1. Log in to your Windows instance using Remote Desktop. For more information, see [Connect to your Windows instance](#).
2. Determine the disk number that you want to make available:
   1. Open the Start menu, and select Windows PowerShell.
   2. Use the `Get-Disk` Cmdlet to retrieve a list of available disks.
   3. In the command output, note the **Number** corresponding to the disk that you're making available.
3. Create a script file to execute DiskPart commands:
   1. Open the Start menu, and select **File Explorer**.
   2. Navigate to a directory, such as C:\, to store the script file.
   3. Choose or right-click an empty space within the folder to open the dialog box, position the cursor over **New** to access the context menu, and then choose **Text Document**.
   4. Name the text file `diskpart.txt`.
4. Add the following commands to the script file. You may need to modify the disk number, partition type, volume label, and drive letter. The script performs the following actions by default:

1. Selects disk 1 for modification.
2. Configures the volume to use the master boot record (MBR) partition structure.
3. Formats the volume as an NTFS volume.
4. Sets the volume label.
5. Assigns the volume a drive letter.

> **Warning**
>
> If you're mounting a volume that already has data on it, do not reformat the volume or you will delete the existing data.

```
select disk 1
attributes disk clear readonly
online disk noerr
convert mbr
create partition primary
format quick fs=ntfs label="volume_label"
assign letter="drive_letter"
```

For more information, see DiskPart Syntax and Parameters.

5. Open a command prompt, navigate to the folder in which the script is located, and run the following command to make a volume available for use on the specified disk:

```
C:\> diskpart /s diskpart.txt
```

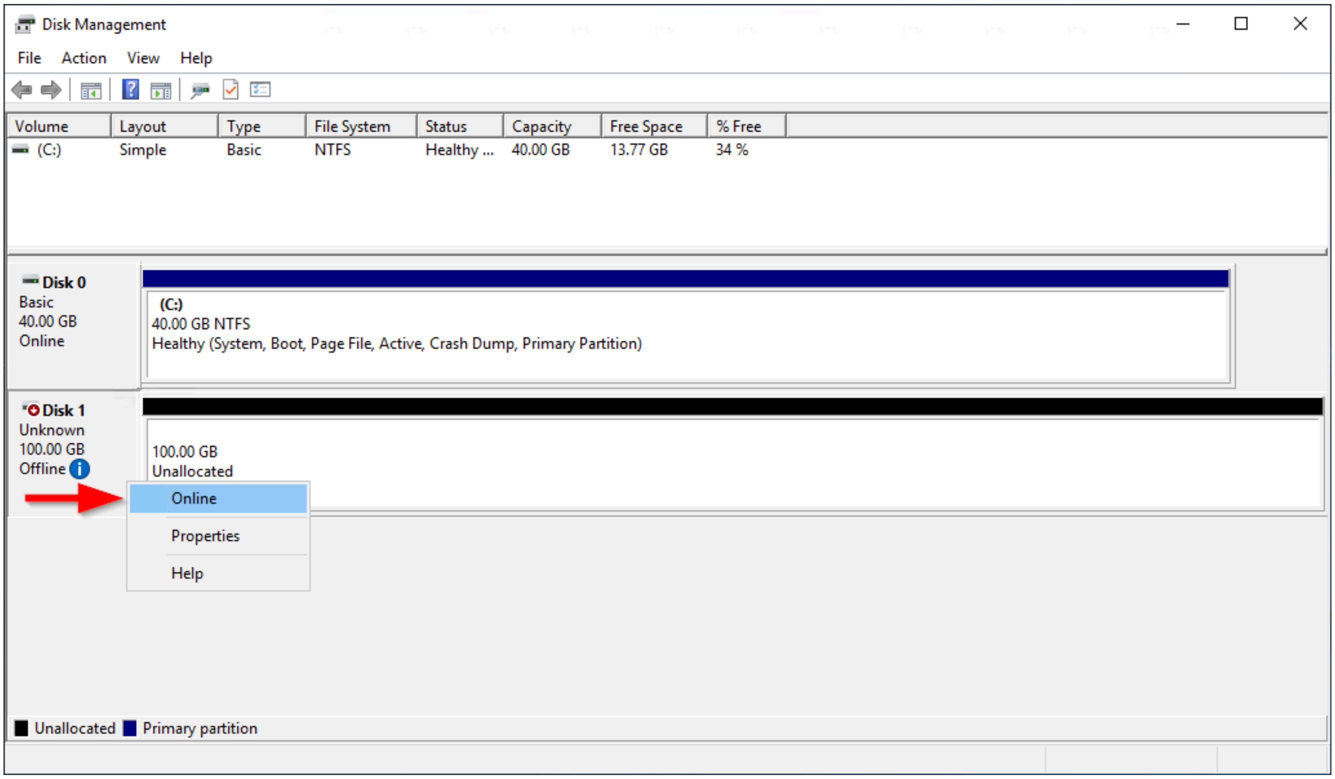
Disk Management utility

To make an EBS volume available to use with the Disk Management utility

1. Log in to your Windows instance using Remote Desktop. For more information, see Connect to your Windows instance.
2. Start the Disk Management utility. On the taskbar, open the context (right-click) menu for the Windows logo, and choose **Disk Management**.

> **Note**
>
> In Windows Server 2008, choose **Start**, **Administrative Tools**, **Computer Management**, **Disk Management**.

3. Bring the volume online. In the lower pane, open the context (right-click) menu for the left panel for the disk for the EBS volume. Choose **Online**.

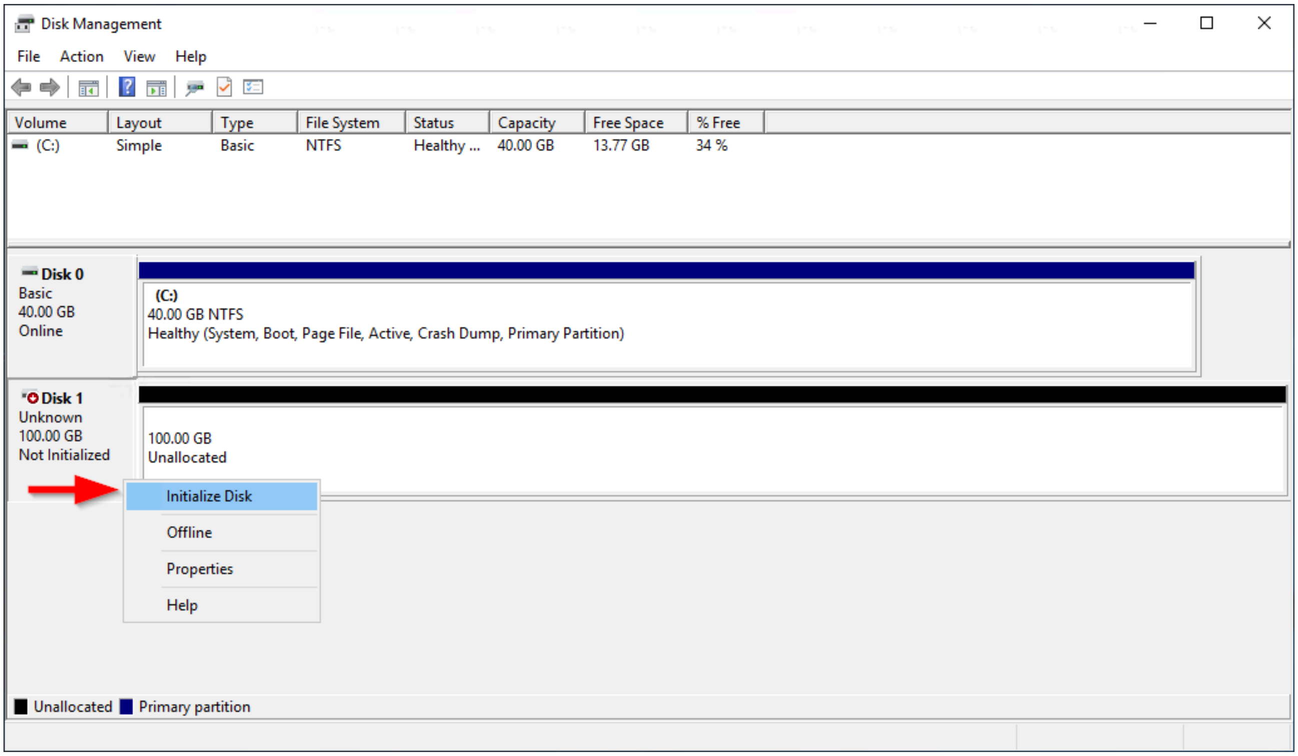


4. (Conditional) If the disk is not initialized, you must initialize it before you can use it. If the disk is already initialized, skip this step.

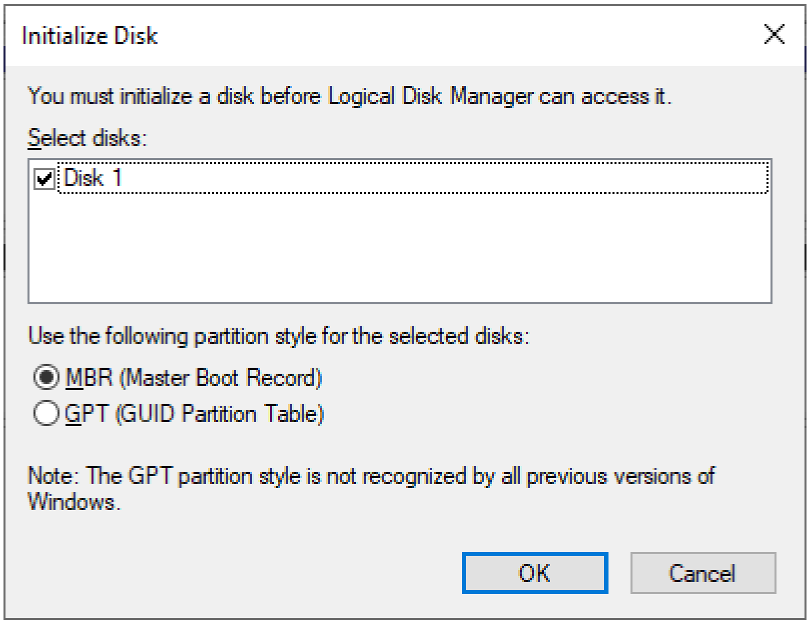
> **Warning**
>
> If you're mounting a volume that already has data on it (for example, a public data set, or a volume that you created from a snapshot), do not reformat the volume or you will delete the existing data.

If the disk is not initialized, initialize it as follows:

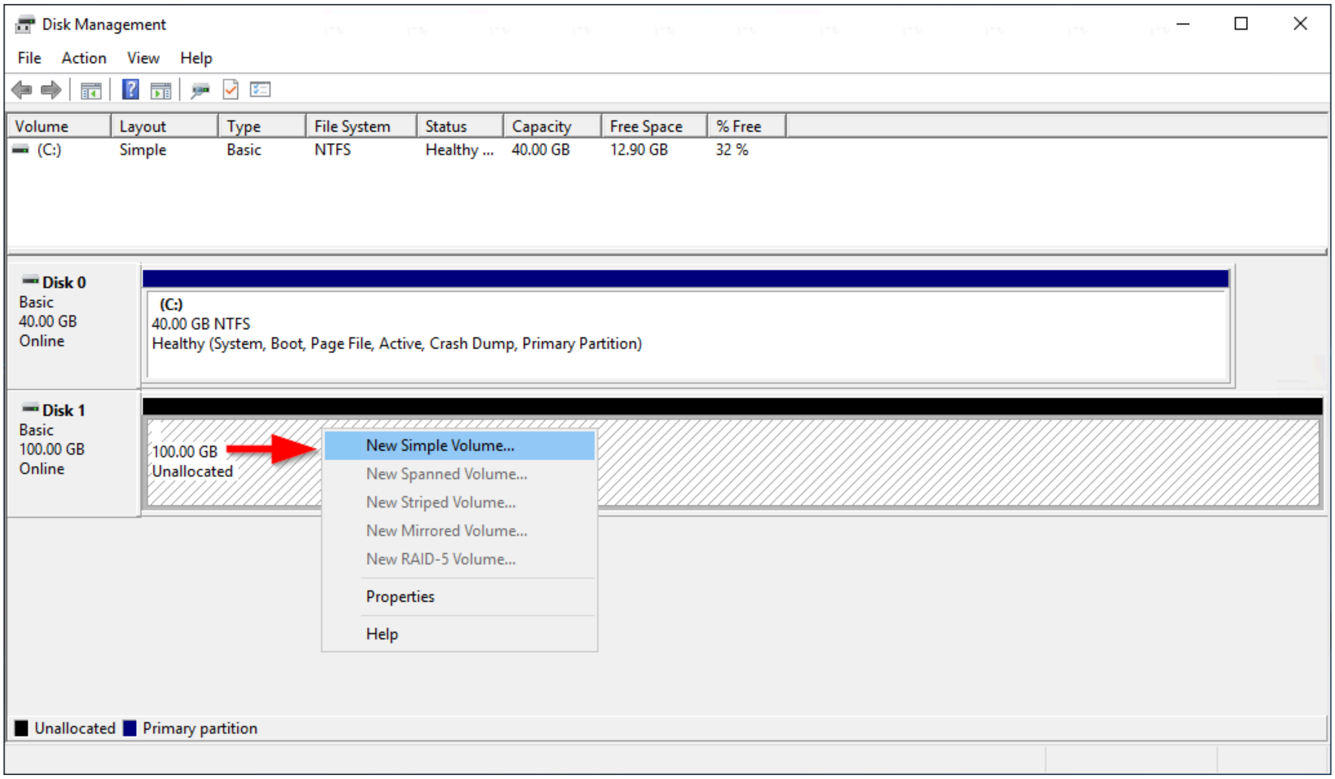
1. Open the context (right-click) menu for the left panel for the disk, and choose **Initialize Disk**.



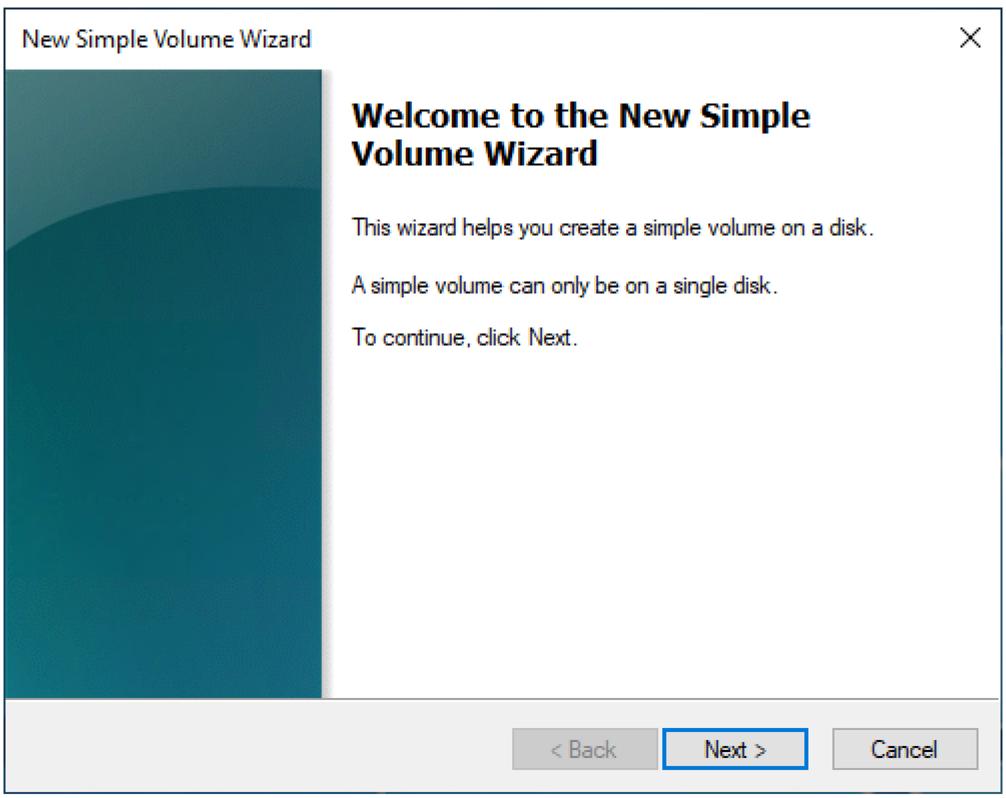
2. In the **Initialize Disk** dialog box, select a partition style, and choose **OK**.



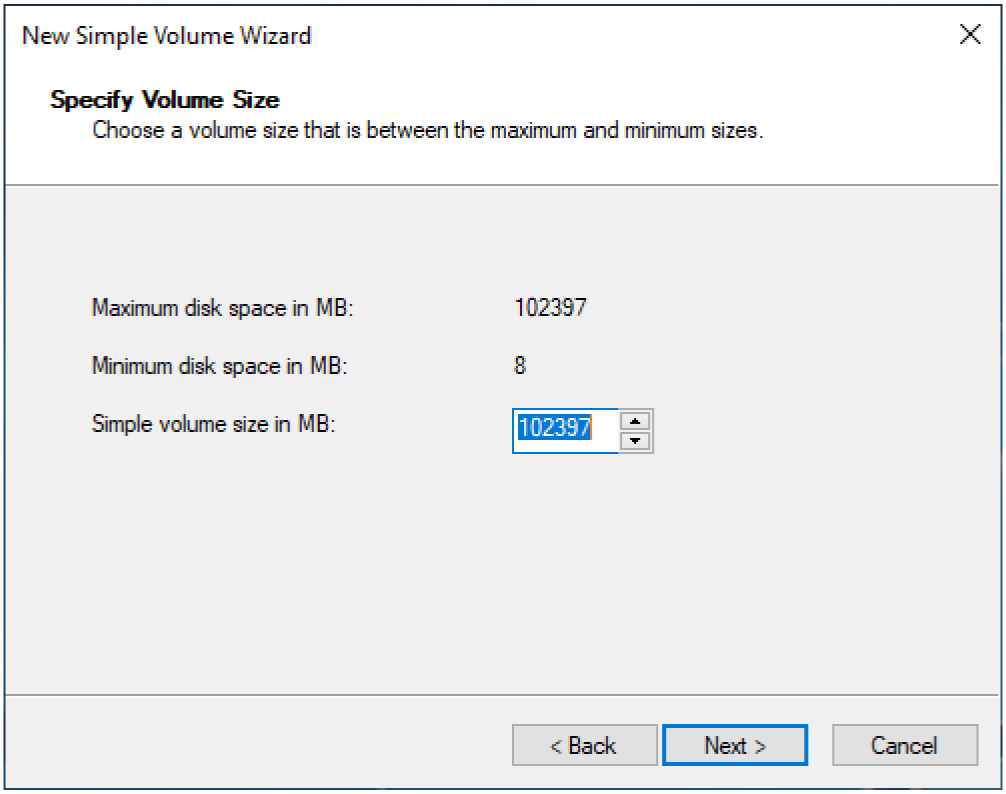
5. Open the context (right-click) menu for the right panel for the disk, and choose **New Simple Volume**.



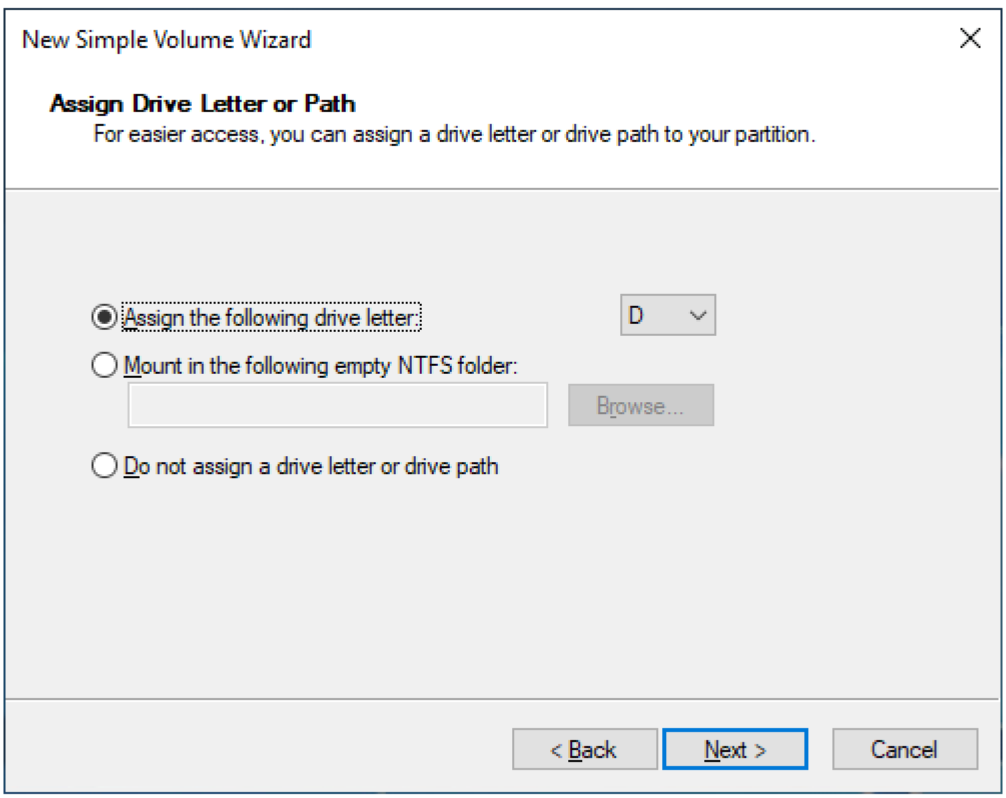
6. In the **New Simple Volume Wizard**, choose **Next**.



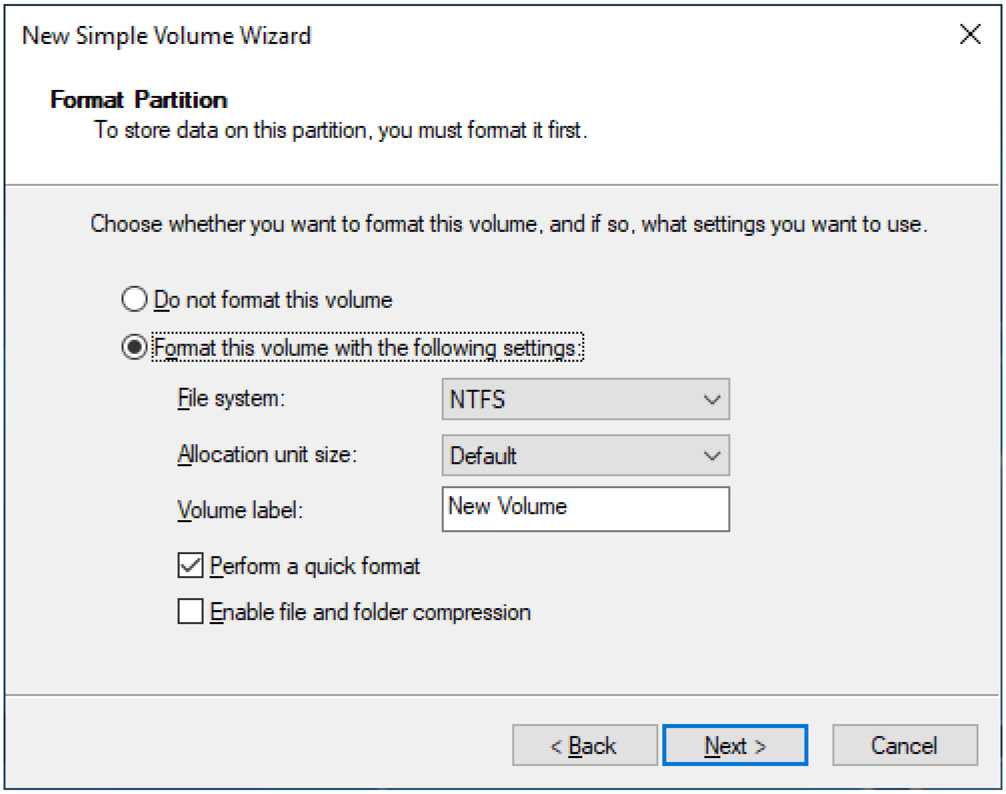
7. If you want to change the default maximum value, specify the **Simple volume size in MB**, and then choose **Next.**



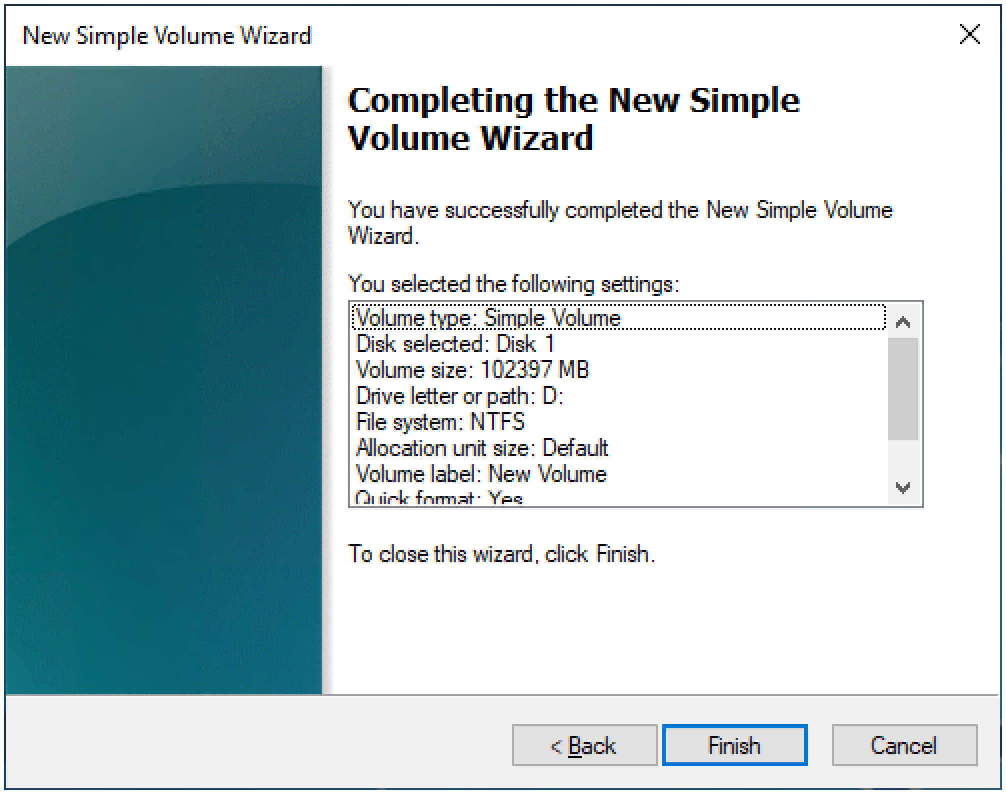
8. Specify a preferred drive letter, if necessary, within the **Assign the following drive letter** dropdown, and then choose **Next.**



9. Specify a **Volume Label** and adjust the default settings as necessary, and then choose **Next.**



10. Review your settings, and then choose **Finish** to apply the modifications and close the New Simple Volume wizard.



View information about an Amazon EBS volume

You can view descriptive information about your EBS volumes. For example, you can view information about all volumes in a specific Region or view detailed information about a single volume, including its size, volume type, whether the volume is encrypted, which KMS key was used to encrypt the volume, and the specific instance to which the volume is attached.

You can get additional information about your EBS volumes, such as how much disk space is available, from the operating system on the instance.

Contents

View volume information

You can view information about your EBS volumes.

Console

To view information about a volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. To reduce the list, you can filter your volumes using tags and volume attributes. Choose the filter field, select a tag or volume attribute, and then select the filter value.
4. To view more information about a volume, choose its ID.

To view the EBS volumes that are attached to an instance

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Instances**.
3. Select the instance.
4. On the **Storage** tab, the **Block devices** section lists the volumes that are attached to the instance. To view information about a specific volume, choose its ID in the **Volume ID** column.

Amazon EC2 Global View

You can use Amazon EC2 Global View to view your volumes across all Regions for which your AWS account is enabled.

To get a summary of your EBS volumes across all Regions

1. Open the Amazon EC2 Global View console at https://console.aws.amazon.com/ec2globalview/home.
2. On the **Region explorer** tab, under **Summary**, check the resource count for **Volumes**, which includes the number of volumes and the number of Regions. Click the underlined text to see how the volume count is spread across Regions.
3. On the **Global search** tab, select the client filter **Resource type = Volume**. You can filter the results further by specifying a Region or a tag.

AWS CLI

To view information about an EBS volume

Use the describe-volumes command. The following example counts the volumes in the current Region.

```
aws ec2 describe-volumes --query "length(Volumes[*])"
```

The following example lists the volumes attached to the specified instance.

```
aws ec2 describe-volumes \
    --filters "Name=attachment.instance-id,Values=i-1234567890abcdef0" \
    --query Volumes[*].VolumeId \
    --output text
```

The following example describes the specified volume.

```
aws ec2 describe-volumes --volume-ids vol-01234567890abcdef
```

The following is example output.

```
{
    "Volumes": [
        {
            "Iops": 3000,
            "VolumeType": "gp3",
            "MultiAttachEnabled": false,
            "Throughput": 125,
            "Operator": {
                "Managed": false
            },
            "VolumeId": "vol-01234567890abcdef",
            "Size": 8,
            "SnapshotId": "snap-0abcdef1234567890",
            "AvailabilityZone": "us-west-2b",
            "State": "in-use",
            "CreateTime": "2024-05-17T23:23:00.400000+00:00",
            "Attachments": [
                {
                    "DeleteOnTermination": true,
```

```
                    "VolumeId": "vol-01234567890abcdef",
                    "InstanceId": "i-1234567890abcdef0",
                    "Device": "/dev/xvda",
                    "State": "attached",
                    "AttachTime": "2024-05-17T23:23:00+00:00"
                }
            ],
            "Encrypted": false
        }
    ]
}
```

PowerShell

To view information about an EBS volume

Use the Get-EC2Volume cmdlet. The following example counts the volumes in the current Region.

```
(Get-EC2Volume).Count
```

The following example lists the volumes attached to the specified instance.

```
(Get-EC2Volume `
    -Filters @{Name="attachment.instance-id";Values="i-1234567890abcdef0"}).VolumeId
```

The following example describes the specified volume.

```
Get-EC2Volume -VolumeId vol-01234567890abcdef
```

The following is example output.

```
Attachments        : {i-1234567890abcdef0}
AvailabilityZone   : us-west-2b
CreateTime         : 5/17/2024 11:23:00 PM
Encrypted          : False
FastRestored       : False
Iops               : 3000
KmsKeyId           :
MultiAttachEnabled : False
Operator           : Amazon.EC2.Model.OperatorResponse
```

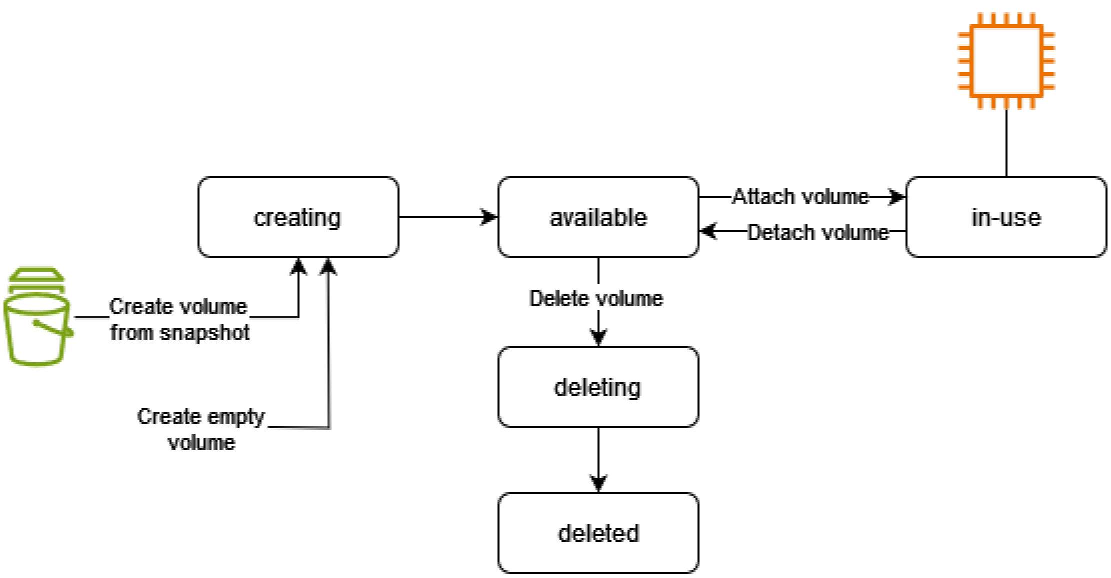
```
OutpostArn          :
Size                : 8
SnapshotId          : snap-0abcdef1234567890
SseType             :
State               : in-use
Tags                : {}
Throughput          : 125
VolumeId            : vol-01234567890abcdef
VolumeType          : gp3
```

Volume states

Volume state describes the availability of an Amazon EBS volume. You can view the volume state in the **State** column on the **Volumes** page in the console, or by using the describe-volumes AWS CLI command.

An Amazon EBS volume transitions through different states from the moment it is created until it is deleted.

The following illustration shows the transitions between volume states. You can create a volume from an Amazon EBS snapshot or create an empty volume. When you create a volume, it enters the `creating` state. After the volume is ready for use, it enters the `available` state. You can attach an available volume to an instance in the same Availability Zone as the volume. You must detach the volume before you can attach it to a different instance or delete it. You can delete a volume when you no longer need it.



The following table summarizes the volume states.

State	Description
`creating`	The volume is being created.
`available`	The volume is not attached to an instance.
`in-use`	The volume is attached to an instance.
`deleting`	The volume is being deleted.
`deleted`	The volume is deleted.
`error`	The underlying hardware related to your EBS volume has failed, and the data associated with the volume is unrecoverable. For information about how to restore the volume or recover the data on the volume, see Why does my EBS volume have a status of "error"?.

View volume metrics

You can get additional information about your EBS volumes from Amazon CloudWatch. For more information, see Amazon CloudWatch metrics for Amazon EBS.

View free disk space

You can get additional information about your EBS volumes, such as how much disk space is available, from the operating system on the instance.

Linux instances

Use the **df -hT** command and specify the device name:

```
[ec2-user ~]$ df -hT /dev/xvda1
Filesystem     Type      Size  Used Avail Use% Mounted on
/dev/xvda1     xfs       8.0G  1.2G  6.9G  15% /
```

Windows instances

You can view the free disk space by opening File Explorer and selecting **This PC**.

You can also view the free disk space using the following `dir` command and examining the last line of the output:

```
C:\> dir C:
 Volume in drive C has no label.
 Volume Serial Number is 68C3-8081

 Directory of C:\

03/25/2018  02:10 AM    <DIR>          .
03/25/2018  02:10 AM    <DIR>          ..
03/25/2018  03:47 AM    <DIR>          Contacts
03/25/2018  03:47 AM    <DIR>          Desktop
03/25/2018  03:47 AM    <DIR>          Documents
03/25/2018  03:47 AM    <DIR>          Downloads
03/25/2018  03:47 AM    <DIR>          Favorites
03/25/2018  03:47 AM    <DIR>          Links
03/25/2018  03:47 AM    <DIR>          Music
03/25/2018  03:47 AM    <DIR>          Pictures
03/25/2018  03:47 AM    <DIR>          Saved Games
03/25/2018  03:47 AM    <DIR>          Searches
03/25/2018  03:47 AM    <DIR>          Videos
               0 File(s)              0 bytes
              13 Dir(s)  18,113,662,976 bytes free
```

You can also view the free disk space using the following `fsutil` command:

```
C:\> fsutil volume diskfree C:
Total # of free bytes        : 18113204224
Total # of bytes             : 32210153472
Total # of avail free bytes  : 18113204224
```

Tip

You can also use the CloudWatch agent to collect disk space usage metrics from an Amazon EC2 instance without connecting to the instance. For more information, see Create the CloudWatch agent configuration file and Installing the CloudWatch agent in the *Amazon CloudWatch User Guide*. If you need to monitor disk space usage for multiple instances, you can install and configure the CloudWatch agent on those instances using Systems Manager. For more information, see Installing the CloudWatch agent using Systems Manager.

Modify an Amazon EBS volume using Elastic Volumes operations

With Amazon EBS Elastic Volumes, you can increase the volume size, change the volume type, or adjust the performance of your EBS volumes. If your instance supports Elastic Volumes, you can do so without detaching the volume or restarting the instance. This enables you to continue using your application while the changes take effect.

There is no charge to modify the configuration of a volume. You are charged for the new volume configuration after volume modification starts. For more information, see the Amazon EBS Pricing page.

Contents

Considerations

- After you initiate a volume modification, you must wait for that modification to reach the `completed` state before you can initiate another modification for the same volume. You can modify a volume up to four times within a rolling 24-hour period, as long as the volume is in the `in-use` or `available` state, and all previous modifications for that volume are `completed`. If you exceed this limit, you get an error message that indicates when you can perform your next modification.
- Volume modifications are performed on a best-effort basis, and they can take from a few minutes to a few hours to complete, depending on the requested volume configuration. Typically, A 1-TiB volume can take up to six hours to be modified. However, the time does not always scale linearly with the volume size - a larger volume might take less time, and a smaller volume might take more time.
- Size increases take effect once the volume modification reaches the `optimizing` state, which usually takes a few seconds.

- Modification time is increased for volumes that are not fully initialized. For more information see Manually initialize the volumes after creation.
- If you change the volume type from gp2 to gp3, and you do not specify IOPS or throughput performance, Amazon EBS automatically provisions either equivalent performance to that of the source gp2 volume, or the baseline gp3 performance, whichever is higher.

  For example, if you modify a 500 GiB gp2 volume with 250 MiB/s throughput and 1500 IOPS to gp3 without specifying IOPS or throughput performance, Amazon EBS automatically provisions the gp3 volume with 3000 IOPS (baseline gp3 IOPS) and 250 MiB/s (to match the source gp2 volume throughput).
- If you encounter an error message while attempting to modify an EBS volume, or if you are modifying an EBS volume attached to a previous-generation instance type, take one of the following steps:
  - For a non-root volume, detach the volume from the instance, apply the modifications, and then re-attach the volume.
  - For a root volume, stop the instance, apply the modifications, and then restart the instance.

Limitations

- You can't cancel a volume modification request after it has been submitted.
- You must increase the volume size. You can't decrease the volume size. However, you can create a smaller volume and then migrate your data to it using an application-level tool such as **rsync** (Linux instances) or **robocopy** (Windows instances).
- There are limits to the maximum aggregated storage that can be requested across volume modifications. For more information, see Amazon EBS service quotas in the *Amazon Web Services General Reference*.
- The new volume size can't exceed the supported capacity of its file system and partitioning scheme. For more information, see Amazon EBS volume constraints.
- If you are not changing the volume type, then volume size and performance modifications must be within the limits of the current volume type. If you are changing the volume type, then volume size and performance modifications must be within the limits of the target volume type. For more information, see Amazon EBS volume types
- Nitro-based instances support volumes provisioned with up to 256,000 IOPS. Other instance types can be attached to volumes provisioned with up to 64,000 IOPS, but can achieve up to 32,000 IOPS.

- You can't modify the volume type for Multi-Attach enabled `io2` volumes.
- You can't modify the volume type, size, or Provisioned IOPS of Multi-Attach enabled `io1` volumes.
- A root volume of type `io1`, `io2`, `gp2`, `gp3`, or `standard` can't be modified to an `st1` or `sc1` volume, even if it is detached from the instance.
- If the volume was attached before November 3, 2016 23:40 UTC, you must initialize Elastic Volumes support. For more information, see Initializing Elastic Volumes Support.
- While `m3.medium` instances fully support volume modification, `m3.large`, `m3.xlarge`, and `m3.2xlarge` instances might not support all volume modification features.

Requirements for Amazon EBS volume modifications

The following requirements and limitations apply when you modify an Amazon EBS volume. To learn more about the general requirements for EBS volumes, see Amazon EBS volume constraints.

Topics

- Supported instance types
- Operating system

Supported instance types

Elastic Volumes are supported on the following instances:

- All current generation instances
- The following previous-generation instances: C1, C3, C4, G2, I2, M1, M3, M4, R3, and R4

If your instance type does not support Elastic Volumes, see Modify an EBS volume if Elastic Volumes is not supported.

Operating system

The following operating system requirements apply:

Linux

Linux AMIs require a GUID partition table (GPT) and GRUB 2 for boot volumes that are 2 TiB (2,048 GiB) or larger. Many Linux AMIs today still use the MBR partitioning scheme, which only supports boot volume sizes up to 2 TiB. If your instance does not boot with a boot volume larger than 2 TiB, the AMI you are using may be limited to a boot volume size of less than 2 TiB. Non-boot volumes do not have this limitation on Linux instances.

Before attempting to resize a boot volume beyond 2 TiB, you can determine whether the volume is using MBR or GPT partitioning by running the following command on your instance:

```
[ec2-user ~]$ sudo gdisk -l /dev/xvda
```

An Amazon Linux instance with GPT partitioning returns the following information:

```
GPT fdisk (gdisk) version 0.8.10

  Partition table scan:
    MBR: protective
    BSD: not present
    APM: not present
    GPT: present

  Found valid GPT with protective MBR; using GPT.
```

A SUSE instance with MBR partitioning returns the following information:

```
GPT fdisk (gdisk) version 0.8.8

  Partition table scan:
    MBR: MBR only
    BSD: not present
    APM: not present
    GPT: not present
```

Windows

By default, Windows initializes volumes with a Master Boot Record (MBR) partition table. Because MBR supports only volumes smaller than 2 TiB (2,048 GiB), Windows prevents you from resizing MBR volumes beyond this limit. In such a case, the **Extend Volume** option is disabled in the Windows **Disk Management** utility. If you use the AWS Management Console or AWS CLI to create an MBR-partitioned volume that exceeds the size limit, Windows cannot detect or use the additional space.

To overcome this limitation, you can create a new, larger volume with a GUID partition table (GPT) and copy over the data from the original MBR volume.

To create a GPT volume

1. Create a new, empty volume of the desired size in the Availability Zone of the EC2 instance and attach it to your instance.

   > **Note**
   >
   > The new volume must not be a volume restored from a snapshot.

2. Log in to your Windows system and open **Disk Management** (**diskmgmt.exe**).
3. Open the context (right-click) menu for the new disk and choose **Online**.
4. In the **Initialize Disk** window, select the new disk and choose **GPT (GUID Partition Table)**, **OK**.
5. When initialization is complete, copy the data from the original volume to the new volume, using a tool such as robocopy or teracopy.
6. In **Disk Management**, change the drive letters to appropriate values and take the old volume offline.
7. In the Amazon EC2 console, detach the old volume from the instance, reboot the instance to verify that it functions properly, and delete the old volume.

Request Amazon EBS volume modifications

With Elastic Volumes, you can dynamically increase the size, increase or decrease the performance, and change the volume type of your Amazon EBS volumes without detaching them.

Process overview

1. (Optional) Before modifying a volume that contains valuable data, it is a best practice to create a snapshot of the volume in case you need to roll back your changes. For more information, see Create Amazon EBS snapshots.
2. Request the volume modification.
3. Monitor the progress of the volume modification. For more information, see Monitor the progress of Amazon EBS volume modifications.

4. If the size of the volume was modified, extend the volume's file system to take advantage of the increased storage capacity. For more information, see Extend the file system after resizing an Amazon EBS volume.

Contents

Modify an EBS volume using Elastic Volumes

Before you begin, see the following:

- Considerations
- Limitations
- Requirements

Console

To modify an EBS volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume to modify and choose **Actions**, **Modify volume**.
4. The **Modify volume** screen displays the volume ID and the volume's current configuration, including type, size, IOPS, and throughput. Set new configuration values as follows:
   - To modify the type, choose a value for **Volume type**.
   - To modify the size, enter a new value for **Size**.
   - (`gp3`, `io1`, and `io2` only) To modify the IOPS, enter a new value for **IOPS**.
   - (`gp3` only) To modify the throughput, enter a new value for **Throughput**.
5. After you have finished changing the volume settings, choose **Modify**. When prompted for confirmation, choose **Modify**.

6. If you've increased the size of your volume, then you must also extend the volume's partition to make use of the additional storage capacity. For more information, see Extend the file system after resizing an Amazon EBS volume.

7. (*Windows instances only*) If you increase the size of an NVMe volume on an instance that does not have the AWS NVMe drivers, you must reboot the instance to enable Windows to see the new volume size. For more information about installing the AWS NVMe drivers, see AWS NVMe drivers.

AWS CLI

To modify an EBS volume

Use the modify-volume command. For example, if you have a volume of type gp2 with a size of 100 GiB, the following example changes its configuration to a volume of type `io1` with 10,000 IOPS and a size of 200 GiB.

```
aws ec2 modify-volume \
    --volume-id vol-01234567890abcdef \
    --volume-type io1 \
    --iops 10000 \
    --size 200
```

The following is example output.

```
{
    "VolumeModification": {
        "TargetSize": 200,
        "TargetVolumeType": "io1",
        "ModificationState": "modifying",
        "VolumeId": "vol-01234567890abcdef",
        "TargetIops": 10000,
        "StartTime": "2022-01-19T22:21:02.959Z",
        "Progress": 0,
        "OriginalVolumeType": "gp2",
        "OriginalIops": 300,
        "OriginalSize": 100
    }
}
```

If you've increased the size of your volume, then you must also extend the volume's partition to make use of the additional storage capacity. For more information, see Extend the file system after resizing an Amazon EBS volume.

PowerShell

To modify an EBS volume

Use the Edit-EC2Volume cmdlet. For example, if you have a volume of type gp2 with a size of 100 GiB, the following example changes its configuration to a volume of type io1 with 10,000 IOPS and a size of 200 GiB.

```
Edit-EC2Volume `
    -VolumeId vol-01234567890abcdef `
    -VolumeType io1 `
    -Iops 10000 `
    -Size 200
```

If you've increased the size of your volume, then you must also extend the volume's partition to make use of the additional storage capacity. For more information, see Extend the file system after resizing an Amazon EBS volume.

Modify an EBS volume if Elastic Volumes is not supported

If you are using a supported instance type, you can use Elastic Volumes to dynamically modify the size, performance, and volume type of your Amazon EBS volumes without detaching them.

If you cannot use Elastic Volumes but you need to modify the root (boot) volume, you must stop the instance, modify the volume, and then restart the instance.

After the instance has started, you can check the file system size to see if your instance recognizes the larger volume space. On Linux, use the **df -h** command to check the file system size.

```
[ec2-user ~]$ df -h
Filesystem            Size  Used Avail Use% Mounted on
/dev/xvda1            7.9G  943M  6.9G  12% /
tmpfs                 1.9G     0  1.9G   0% /dev/shm
```

If the size does not reflect your newly expanded volume, you must extend the file system of your device so that your instance can use the new space. For more information, see Extend the file system after resizing an Amazon EBS volume.

With Windows instances, you might have to bring the volume online in order to use it. For more information, see Make an Amazon EBS volume available for use. You do not need to reformat the volume.

Initialize Elastic Volumes support (if needed)

Before you can modify a volume that was attached to an instance before November 3, 2016 23:40 UTC, you must initialize volume modification support using one of the following actions:

- Detach and attach the volume
- Stop and start the instance

Console

To determine whether your instances are ready

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. On the navigation pane, choose **Instances**.
3. Choose the **Show/Hide Columns** icon (the gear). Select the **Launch time** attribute column and then choose **Confirm**.
4. Sort the list of instances by the **Launch Time** column. For each instance that was started before the cutoff date, choose the **Storage** tab and check the **Attachment time** column to see when its volumes were attached.

AWS CLI

To determine whether your instances are ready

Use the following describe-instances command to determine whether the volume was attached before November 3, 2016 23:40 UTC.

```
aws ec2 describe-instances \
    --query "Reservations[*].Instances[*].
[InstanceId,LaunchTime<='2016-11-01',BlockDeviceMappings[*]
[Ebs.AttachTime<='2016-11-01']]" \
    --output text
```

The first line of the output for each instance shows its ID and whether it was started before the cutoff date (True or False). The first line is followed by one or more lines that show whether each EBS volume was attached before the cutoff date (True or False). In the following example output, you must initialize volume modification for the first instance because it was started before the cutoff date and its root volume was attached before the cutoff date. The other instances are ready because they were started after the cutoff date.

```
i-e905622e              True
True
i-719f99a8              False
True
i-006b02c1b78381e57     False
False
False
i-e3d172ed              False
True
```

PowerShell

To determine whether an instance is ready

Use the Get-EC2Instance cmdlet to determine whether a volume was attached before November 3, 2016 23:40 UTC.

```
(Get-EC2Instance `
    -InstanceId i-1234567890abcdef0).Instances.BlockDeviceMappings | `
     Format-Table @{Name="VolumeId";Expression={$_.Ebs.VolumeId}}, `
                  @{Name="AttachTime";Expression={$_.Ebs.AttachTime}}
```

The following is example output.

```
VolumeId              AttachTime
--------              ----------
vol-0b243c8d927752d2b 3/23/2020 12:21:14 AM
vol-043eadbeb4a8387c3 9/5/2020 7:39:22 PM
vol-0c3f0c4e55c082753 4/23/2019 4:07:40 PM
```

Monitor the progress of Amazon EBS volume modifications

When you modify an EBS volume, it goes through a sequence of states. The volume enters the `modifying` state, the `optimizing` state, and finally the `completed` state. At this point, the volume is ready to be further modified.

While the volume is in the `optimizing` state, your volume performance is in between the source and target configuration specifications. Transitional volume performance will be no less than the source volume performance. If you are downgrading IOPS, transitional volume performance is no less than the target volume performance.

Volume modification changes take effect as follows:

- Size increases take effect once the volume modification reaches the `optimizing` state, which usually takes a few seconds.
- Performance (IOPS and throughput) changes can take from a few minutes to a few hours to complete, depending on the requested volume configuration. Typically, a fully used 1-TiB volume can take about 6 hours to migrate to a new performance configuration. In some cases, it can take more than 24 hours for a new performance configuration to take effect, such as when the volume has not been fully initialized.

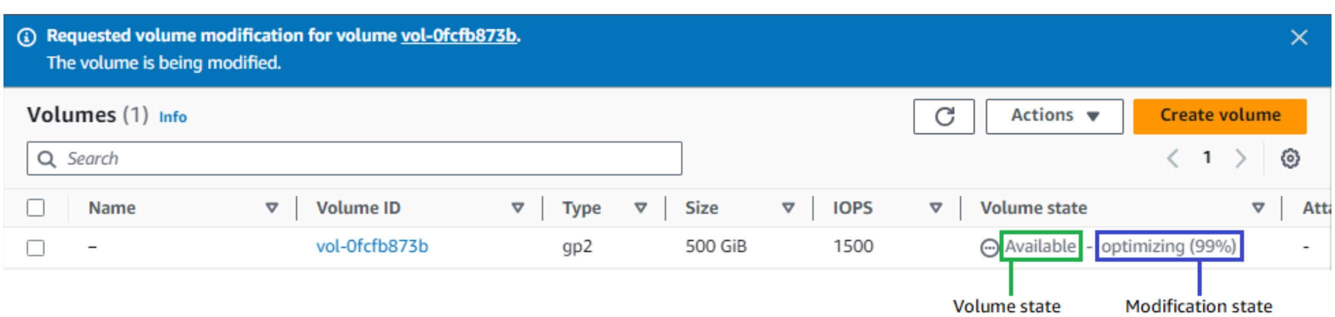
The possible volume states are `creating`, `available`, `in-use`, `deleting`, `deleted`, and `error`.

The possible modification states are `modifying`, `optimizing`, and `completed`.

Console

To monitor progress of a modification

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume.
4. The **Volume state** column and the **Volume state** field in the **Details** tab contain information in the following format: *`Volume state`* - *`Modification state`* (*`Modification progress`*%). The following image shows the volume and volume modification states.



After the modification completes, only the volume state is displayed. The modification state and progress are no longer displayed.

Alternatively, you can use Amazon EventBridge to create a notification rule for volume modification events. For more information, see Getting started with Amazon EventBridge.

AWS CLI

To monitor progress of a modification

Use the describe-volumes-modifications command to view the progress of one or more volume modifications. The following example describes the volume modifications for two volumes.

```
aws ec2 describe-volumes-modifications \
    --volume-ids vol-11111111111111111 vol-22222222222222222
```

In the following example output, the volume modifications are still in the `modifying` state. Progress is reported as a percentage.

```
{
    "VolumesModifications": [
        {
            "TargetSize": 200,
            "TargetVolumeType": "io1",
            "ModificationState": "modifying",
            "VolumeId": "vol-11111111111111111",
            "TargetIops": 10000,
            "StartTime": "2017-01-19T22:21:02.959Z",
            "Progress": 0,
            "OriginalVolumeType": "gp2",
            "OriginalIops": 300,
            "OriginalSize": 100
        },
        {
            "TargetSize": 2000,
            "TargetVolumeType": "sc1",
            "ModificationState": "modifying",
            "VolumeId": "vol-22222222222222222",
            "StartTime": "2017-01-19T22:23:22.158Z",
            "Progress": 0,
            "OriginalVolumeType": "gp2",
```

```
            "OriginalIops": 300,
            "OriginalSize": 1000
        }
    ]
}
```

The next example describes all volumes with a modification state of either `optimizing` or `completed`, and then filters and formats the results to show only modifications that were initiated on or after February 1, 2017:

```
aws ec2 describe-volumes-modifications \
    --filters Name=modification-state,Values="optimizing","completed" \
    --query "VolumesModifications[?StartTime>='2017-02-01'].
{ID:VolumeId,STATE:ModificationState}"
```

The following is example output with information about two volumes:

```
[
    {
        "STATE": "optimizing",
        "ID": "vol-06397e7a0eEXAMPLE"
    },
    {
        "STATE": "completed",
        "ID": "vol-ba74e18c2aEXAMPLE"
    }
]
```

PowerShell

To monitor progress of a modification

Use the Get-EC2VolumeModification cmdlet. The following example describes the volume modifications for two volumes.

```
Get-EC2VolumeModification `
    -VolumeId vol-11111111111111111 vol-22222222222222222
```

> **Note**
>
> Rarely, a transient AWS fault can result in a `failed` state. This is not an indication of volume health; it merely indicates that the modification to the volume failed. If this occurs, retry the volume modification.

Extend the file system after resizing an Amazon EBS volume

After you increase the size of an EBS volume, you must extend the partition and file system to the new, larger size. You can do this as soon as the volume enters the `optimizing` state.

Before you begin

- Create a snapshot of the volume, in case you need to roll back your changes. For more information, see Create Amazon EBS snapshots.
- Confirm that the volume modification succeeded and that it is in the`optimizing` or `completed` state. For more information, see Monitor the progress of Amazon EBS volume modifications.
- Ensure that the volume is attached to the instance and that it is formatted and mounted. For more information, see Format and mount an attached volume.
- (*Linux instances only*) If you are using logical volumes on the Amazon EBS volume, you must use Logical Volume Manager (LVM) to extend the logical volume. For instructions about how to do this, see the **Extend the LV** section in the article How do I use LVM to create a logical volume on an EBS volume's partition?.

Linux instances

> **Note**
>
> The following instructions walk you through the process of extending **XFS** and **Ext4** file systems for Linux. For information about extending a different file system, see its documentation.

Before you can extend a file system on Linux, you must extend the partition, if your volume has one.

Extend the file system of EBS volumes

Use the following procedure to extend the file system for a resized volume.

Note that device and partition naming differs for Xen instances and instances built on the Nitro System. To determine whether your instance is Xen-based or Nitro-based, see Amazon EC2 hypervisor type.

To extend the file system of EBS volumes

1. Connect to your instance.
2. Resize the partition, if needed. To do so:

   a. Check whether the volume has a partition. Use the **lsblk** command.

      Nitro instance example

      In the following example output, the root volume (`nvme0n1`) has two partitions (`nvme0n1p1` and `nvme0n1p128`), while the additional volume (`nvme1n1`) has no partitions.

```
[ec2-user ~]$ sudo lsblk
NAME           MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
nvme1n1        259:0    0  30G  0 disk /data
nvme0n1        259:1    0  16G  0 disk
##nvme0n1p1    259:2    0   8G  0 part /
##nvme0n1p128 259:3    0   1M  0 part
```

      Xen instance example

      In the following example output, the root volume (`xvda`) has a partition (`xvda1`), while the additional volume (`xvdf`) has no partition.

```
[ec2-user ~]$ sudo lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  16G  0 disk
##xvda1 202:1    0   8G  0 part /
xvdf    202:80   0  24G  0 disk
```

      - If the volume has a partition, continue to the next step (2b).

- If the volume has no partitions, skip steps 2b, 2c, and 2d, and continue to step 3.

> **Troubleshooting tip**
>
> If you do not see the volume in the command output, ensure that the volume is attached to the instance, and that it is formatted and mounted.

b. Check whether the partition needs to be extended. In the **lsblk** command output from the previous step, compare the partition size and the volume size.

- If the partition size is smaller than the volume size, continue to the next step (2c).
- If the partition size is equal to the volume size, the partition does not need to be extended - skip steps 2c and 2d, and continue to step 3.

> **Troubleshooting tip**
>
> If the volume still reflects the original size, confirm that the volume modification succeeded.

c. Extend the partition. Use the **growpart** command and specify the device name and the partition number.

Nitro instance example

The partition number is the number after the p. For example, for `nvme0n1p1`, the partition number is 1. For `nvme0n1p128`, the partition number is 128.

To extend a partition named `nvme0n1p1`, use the following command.

> **Important**
>
> Note the space between the device name (`nvme0n1`) and the partition number (1).

```
[ec2-user ~]$ sudo growpart /dev/nvme0n1 1
```

Xen instance example

The partition number is the number after the device name. For example, for `xvda1`, the partition number is `1`. For `xvda128`, the partition number is `128`.

To extend a partition named `xvda1`, use the following command.

> **Important**
>
> Note the space between the device name (`xvda`) and the partition number (1).

```
[ec2-user ~]$ sudo growpart /dev/xvda 1
```

> **Troubleshooting tips**
>
> - `mkdir: cannot create directory '/tmp/growpart.31171': No space left on device FAILED: failed to make temp dir`: Indicates that there is not enough free disk space on the volume for growpart to create the temporary directory it needs to perform the resize. Free up some disk space and then try again.
> - `must supply partition-number`: Indicates that you specified an incorrect partition. Use the **lsblk** command to confirm the partition name, and ensure that you enter a space between the device name and the partition number.
> - `NOCHANGE: partition 1 is size 16773087. it cannot be grown`: Indicates that the partition already extends the entire volume and can't be extended. Confirm that the volume modification succeeded.

d. Verify that the partition has been extended. Use the **lsblk** command. The partition size should now be equal to the volume size.

Nitro instance example

The following example output shows that both the volume (`nvme0n1`) and the partition (`nvme0n1p1`) are the same size (`16 GB`).

```
[ec2-user ~]$ sudo lsblk
NAME          MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
nvme1n1       259:0    0  30G  0 disk /data
nvme0n1       259:1    0  16G  0 disk
##nvme0n1p1   259:2    0  16G  0 part /
##nvme0n1p128 259:3    0   1M  0 part
```

Xen instance example

The following example output shows that both the volume (`xvda`) and the partition (`xvda1`) are the same size (`16 GB`).

```
[ec2-user ~]$ sudo lsblk
NAME    MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvda    202:0    0  16G  0 disk
##xvda1 202:1    0  16G  0 part /
xvdf    202:80   0  24G  0 disk
```

3. Extend the file system.

   a. Get the name, size, type, and mount point for the file system that you need to extend. Use the **df -hT** or **lsblk -f** command.

   Nitro instance example

   The following example output for the **df -hT** command shows that the `/dev/nvme0n1p1` file system is 8 GB in size, its type is `xfs`, and its mount point is `/`.

```
[ec2-user ~]$ df -hT
Filesystem      Type  Size  Used Avail Use% Mounted on
/dev/nvme0n1p1  xfs   8.0G  1.6G  6.5G  20% /
/dev/nvme1n1    xfs   8.0G   33M  8.0G   1% /data
...
```

   Xen instance example

   The following example output for the **df -hT** command shows that the `/dev/xvda1` file system is 8 GB in size, its type is `ext4`, and its mount point is `/`.

```
[ec2-user ~]$ df -hT
```

```
Filesystem      Type   Size    Used   Avail   Use%   Mounted on
/dev/xvda1      ext4   8.0G    1.9G   6.2G    24%    /
/dev/xvdf1      xfs    24.0G   45M    8.0G    1%     /data
...
```

- If the file system size is smaller than the volume size, continue to the next step (3b).
- If the file system size is equal to the volume size, then it does not need to be extended. In this case, skip the remaining steps - the partition and file system have been extended to the new volume size.

b. The commands to extend the file system differ depending on the file system type. Choose the following correct command based on the file system type that you noted in the previous step.

- **[XFS file system]** Use the **xfs_growfs** command and specify the mount point of the file system that you noted in the previous step.

  Nitro and Xen instance example

  For example, to extend a file system mounted on /, use the following command.

  ```
  [ec2-user ~]$ sudo xfs_growfs -d /
  ```

  Troubleshooting tips

  - `xfs_growfs: /data is not a mounted XFS filesystem`: Indicates that you specified the incorrect mount point, or the file system is not XFS. To verify the mount point and file system type, use the **df -hT** command.
  - `data size unchanged, skipping`: Indicates that the file system already extends the entire volume. If the volume has no partitions, confirm that the volume modification succeeded. If the volume has partitions, ensure that the partition was extended as described in step 2.

- **[Ext4 file system]** Use the **resize2fs** command and specify the name of the file system that you noted in the previous step.

Nitro instance example

For example, to extend a file system mounted named `/dev/nvme0n1p1`, use the following command.

```
[ec2-user ~]$ sudo resize2fs /dev/nvme0n1p1
```

Xen instance example

For example, to extend a file system mounted named `/dev/xvda1`, use the following command.

```
[ec2-user ~]$ sudo resize2fs /dev/xvda1
```

Troubleshooting tips

- `resize2fs: Bad magic number in super-block while trying to open /dev/xvda1`: Indicates that the file system is not Ext4. To verify file the system type, use the **df -hT** command.
- `open: No such file or directory while opening /dev/xvdb1`: Indicates that you specified an incorrect partition. To verify the partition, use the **df -hT** command.
- `The filesystem is already 3932160 blocks long. Nothing to do!`: Indicates that the file system already extends the entire volume. If the volume has no partitions, confirm that the volume modification succeeded. If the volume has partitions, ensure that the partition was extended, as described in step 2.

- **[Other file system]** See the documentation for your file system for instructions.

c. Verify that the file system has been extended. Use the **df -hT** command and confirm that the file system size is equal to the volume size.

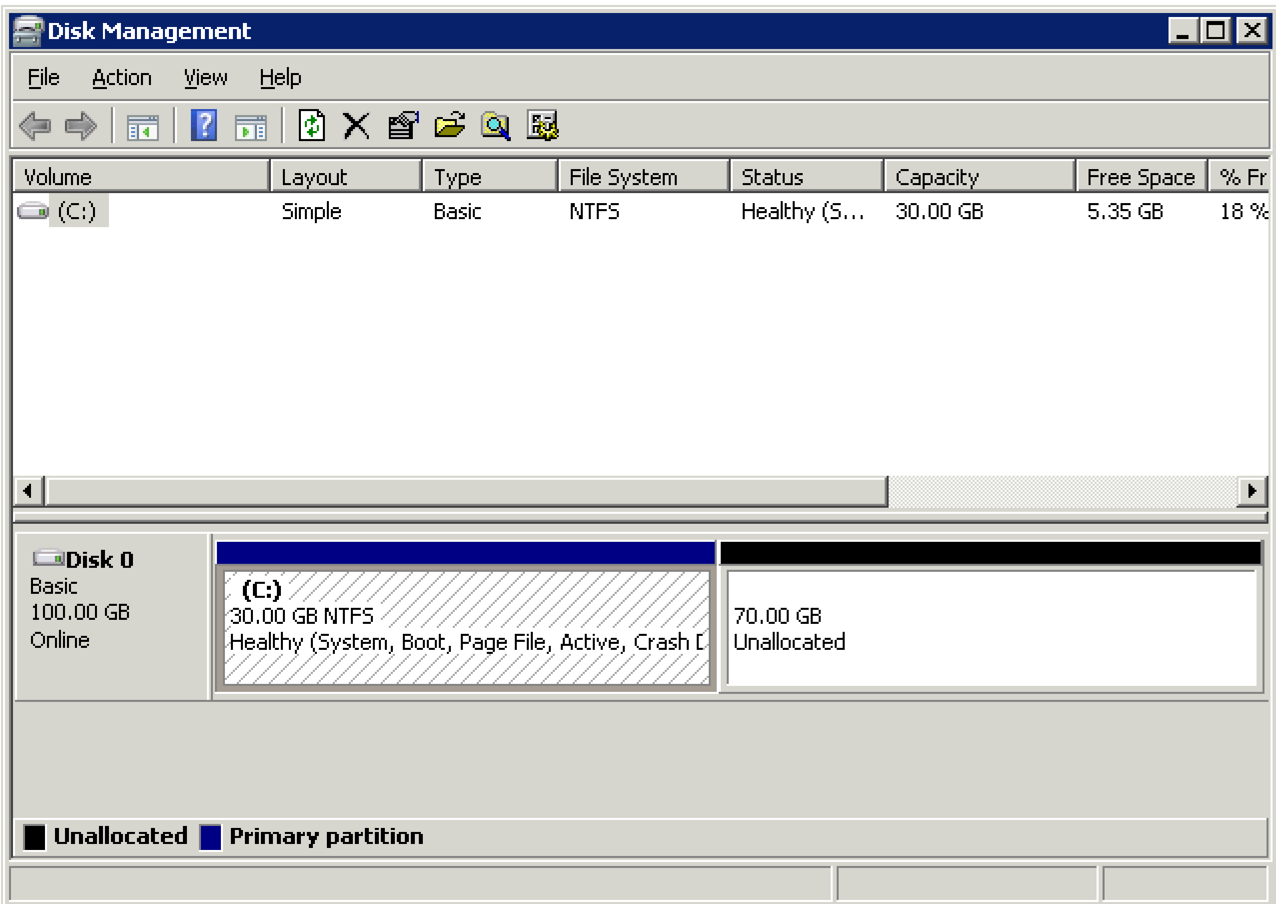
Windows instances

Use one of the following methods to extend the file system on a Windows instance.

Disk Management utility

To extend a file system using Disk Management

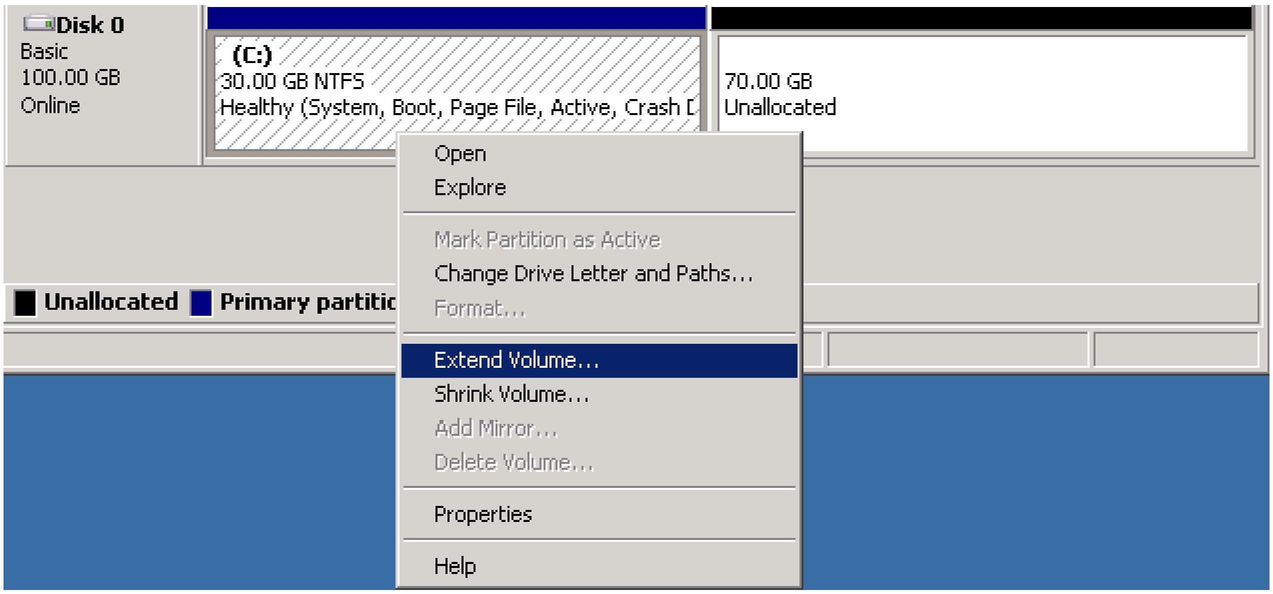
1. Before extending a file system that contains valuable data, it is a best practice to create a snapshot of the volume that contains it in case you need to roll back your changes. For more information, see Create Amazon EBS snapshots.
2. Log in to your Windows instance using Remote Desktop.
3. In the **Run** dialog, enter **diskmgmt.msc** and press Enter. The Disk Management utility opens.



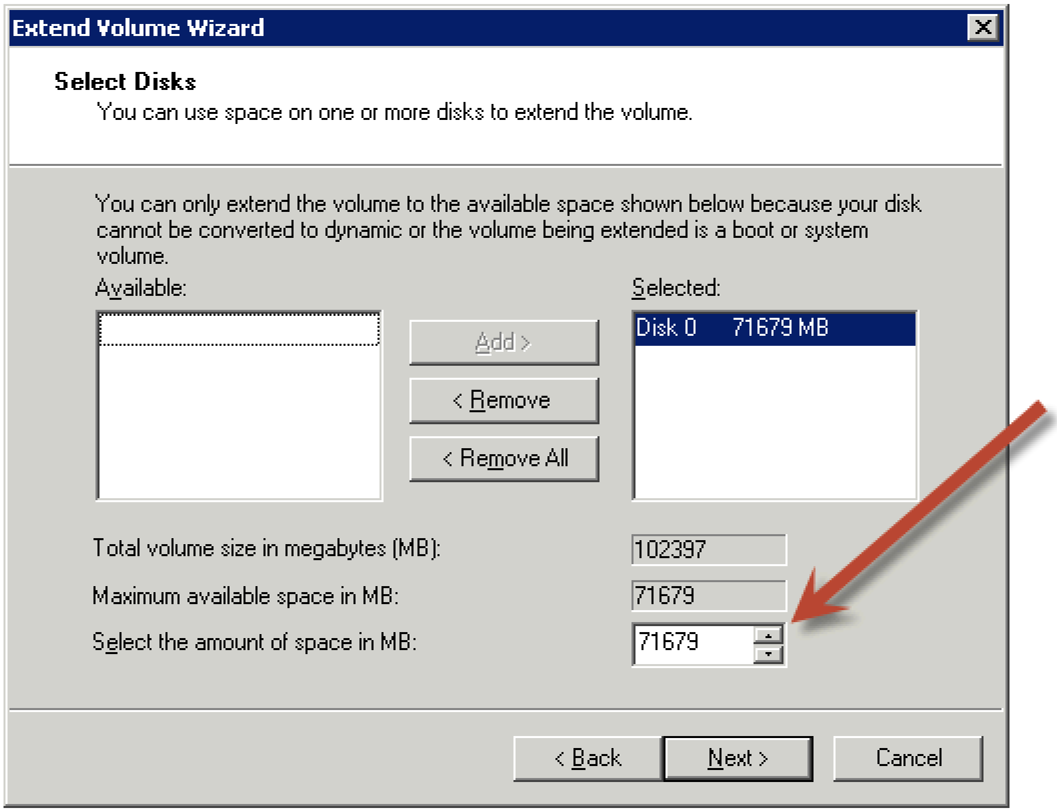
4. On the **Disk Management** menu, choose **Action**, **Rescan Disks**.
5. Open the context (right-click) menu for the expanded drive and choose **Extend Volume**.

> **Note**
>
> **Extend Volume** might be disabled (grayed out) if:

- The unallocated space is not adjacent to the drive. The unallocated space must be adjacent to the right side of the drive you want to extend.
- The volume uses the Master Boot Record (MBR) partition style and it is already 2TB in size. Volumes that use MBR cannot exceed 2TB in size.



6. In the **Extend Volume** wizard, choose **Next**. For **Select the amount of space in MB**, enter the number of megabytes by which to extend the volume. Generally, you specify the maximum available space. The highlighted text under **Selected** is the amount of space that is added, not the final size the volume will have. Complete the wizard.



7. If you increase the size of an NVMe volume on an instance that does not have the AWS NVMe driver, you must reboot the instance to enable Windows to see the new volume size. For more information about installing the AWS NVMe driver, see AWS NVMe drivers.

PowerShell

Use the following procedure to extend a Windows file system using PowerShell.

To extend a file system using PowerShell

1. Before extending a file system that contains valuable data, it is a best practice to create a snapshot of the volume that contains it in case you need to roll back your changes. For more information, see Create Amazon EBS snapshots.
2. Log in to your Windows instance using Remote Desktop.
3. Run PowerShell as an administrator.
4. Run the `Get-Partition` command. PowerShell returns the corresponding partition number for each partition, the drive letter, offset, size, and type. Note the drive letter of the partition to extend.
5. Run the following command to rescan the disk.

```
"rescan" | diskpart
```

6. Run the following command, using the drive letter you noted in step 4 in place of **`<drive-letter>`**. PowerShell returns the minimum and maximum size of the partition allowed, in bytes.

```
Get-PartitionSupportedSize -DriveLetter <drive-letter>
```

7. To extend the partition to a specified amount, run the following command, entering the new size of the volume in place of **`<size>`**. You can enter the size in `KB`, `MB`, and `GB`; for example, `50GB`.

```
Resize-Partition -DriveLetter <drive-letter> -Size <size>
```

   To extend the partition to the maximum available size, run the following command.

```
Resize-Partition -DriveLetter <drive-letter> -Size $(Get-PartitionSupportedSize
 -DriveLetter <drive-letter>).SizeMax
```

   The following PowerShell commands show the complete command and response flow for extending a file system to a specific size.

```
PS C:\> Get-Partition


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&26a12046&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                C           1048576                                      30 GB IFS


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&34763423&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                D           1048576                                       8 MB IFS


PS C:\> "rescan" | diskpart

Microsoft DiskPart version 10.0.17763.1911

Copyright (C) Microsoft Corporation.
On computer:

DISKPART>
Please wait while DiskPart scans your configuration...

DiskPart has finished scanning your configuration.

DISKPART>
PS C:\> Get-PartitionSupportedSize -DriveLetter D

SizeMin      SizeMax
-------      -------
8388608 107372085248


PS C:\> Resize-Partition -DriveLetter D -Size 50GB
PS C:\> Get-Partition


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&26a12046&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                C           1048576                                      30 GB IFS


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&34763423&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                D           1048576                                      50 GB IFS
```

The following PowerShell commands show the complete command and response flow for extending a file system to the maximum available size.

```
PS C:\> Get-Partition


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&26a12046&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                C           1048576                                      30 GB IFS


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&34763423&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                D           1048576                                      50 GB IFS


PS C:\> "rescan" | diskpart

Microsoft DiskPart version 10.0.17763.1911

Copyright (C) Microsoft Corporation.
On computer:

DISKPART>
Please wait while DiskPart scans your configuration...

DiskPart has finished scanning your configuration.

DISKPART>
PS C:\> Get-PartitionSupportedSize -DriveLetter D

 SizeMin      SizeMax
 -------      -------
59047936 107372085248


PS C:\> Resize-Partition -DriveLetter D -Size $(Get-PartitionSupportedSize -DriveLetter D).SizeMax
PS C:\> Get-Partition


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&26a12046&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                C           1048576                                      30 GB IFS


   DiskPath: \\?\scsi#disk&ven_nvme&prod_amazon_elastic_b#4&34763423&0&000000#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}

PartitionNumber  DriveLetter Offset                                        Size Type
---------------  ----------- ------                                        ---- ----
1                D           1048576                                     100 GB IFS
```

Detach an Amazon EBS volume from an Amazon EC2 instance

You need to detach an Amazon Elastic Block Store (Amazon EBS) volume from an instance before you can attach it to a different instance or delete it. Detaching a volume does not affect the data on the volume.

Topics

- Considerations
- Unmount and detach a volume

- Troubleshoot

Considerations

- You can detach an Amazon EBS volume from an instance explicitly or by terminating the instance. However, if the instance is running, you must first unmount the volume from the instance.
- If an EBS volume is the root device of an instance, you must stop the instance before you can detach the volume.
- You can reattach a volume that you detached (without unmounting it), but it might not get the same mount point. If there were writes to the volume in progress when it was detached, the data on the volume might be out of sync.
- After you detach a volume, you are still charged for volume storage as long as the storage amount exceeds the limit of the AWS Free Tier. You must delete a volume to avoid incurring further charges. For more information, see Delete an Amazon EBS volume.

Unmount and detach a volume

Use the following procedures to unmount and detach a volume from an instance. This can be useful when you need to attach the volume to a different instance or when you need to delete the volume.

Steps

- Step 1: Unmount the volume
- Step 2: Detach the volume from the instance
- Step 3: (Windows instances only) Uninstall the offline device locations

Step 1: Unmount the volume

Linux instances

From your Linux instance, use the following command to unmount the `/dev/sdh` device.

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

Windows instances

From your Windows instance, unmount the volume as follows.

1. Start the Disk Management utility.
   - (Windows Server 2012 and later) On the taskbar, right-click the Windows logo and choose **Disk Management**.
   - Windows Server 2008) Choose **Start**, **Administrative Tools**, **Computer Management**, **Disk Management**.
2. Right-click the disk (for example, right-click **Disk 1**) and then choose **Offline**. Wait for the disk status to change to **Offline** before opening the Amazon EC2 console.

Step 2: Detach the volume from the instance

To detach the volume from the instance, use one of the following methods:

Console

To detach an EBS volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume.
4. Choose **Actions**, **Detach volume**.
5. When prompted for confirmation, choose **Detach**.

AWS CLI

To detach an EBS volume from an instance

After unmounting the volume, use the detach-volume command.

```
aws ec2 detach-volume --volume-id vol-01234567890abcdef
```

PowerShell

To detach an EBS volume from an instance

After unmounting the volume, use the [Dismount-EC2Volume] cmdlet.

```
Dismount-EC2Volume -VolumeId vol-01234567890abcdef
```

Step 3: (*Windows instances only*) Uninstall the offline device locations

When you unmount and detach a volume from an instance, Windows flags the device location as offline. The device location remains offline after rebooting, and stopping and restarting the instance. When you restart the instance, Windows might mount one of the remaining volumes to the offline device location. This causes the volume to be unavailable in Windows. To prevent this from happening and to ensure that all volumes are attached to online device locations the next time Windows starts, perform the following steps:

1. On the instance, open the Device Manager.
2. In the Device Manager, select **View**, **Show hidden devices**.
3. In the list of devices, expand the **Storage controllers** node.

   The device locations to which the detached volumes were mounted are named `AWS NVMe Elastic Block Storage Adapter` and they should appear greyed out.
4. Right-click each greyed out device location named `AWS NVMe Elastic Block Storage Adapter`, select **Uninstall device** and choose **Uninstall**.

   > **Important**
   >
   > Do not select the **Delete the driver software for this device** check box.

Troubleshoot

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

> **Note**
>
> To guard against the possibility of data loss, take a snapshot of your volume before attempting to unmount it. Forced detachment of a stuck volume can cause damage to the

file system or the data it contains or an inability to attach a new volume using the same device name, unless you reboot the instance.

- If you encounter problems while detaching a volume through the Amazon EC2 console, it can be helpful to use the **describe-volumes** CLI command to diagnose the issue. For more information, see describe-volumes.
- If your volume stays in the `detaching` state, you can force the detachment by choosing **Force Detach**. Use this option only as a last resort to detach a volume from a failed instance, or if you are detaching a volume with the intention of deleting it. The instance doesn't get an opportunity to flush file system caches or file system metadata. If you use this option, you must perform the file system check and repair procedures.
- If you've tried to force the volume to detach multiple times over several minutes and it stays in the `detaching` state, you can post a request for help to AWS re:Post. To help expedite a resolution, include the volume ID and describe the steps that you've already taken.
- When you attempt to detach a volume that is still mounted, the volume can become stuck in the `busy` state while it is trying to detach. The following output from **describe-volumes** shows an example of this condition:

```
"Volumes": [
    {
        "AvailabilityZone": "us-west-2b",
        "Attachments": [
            {
                "AttachTime": "2022-07-21T23:44:52.000Z",
                "InstanceId": "i-1234567890abcdef0",
                "VolumeId": "vol-01234567890abcdef",
                "State": "busy",
                "DeleteOnTermination": false,
                "Device": "/dev/sdf"
            }
        ...
    }
]
```

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

Delete an Amazon EBS volume

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

You can't delete a volume if it's attached to an instance. To delete a volume, you must first detach it. For more information, see Detach an Amazon EBS volume from an Amazon EC2 instance.

If you delete a volume that matches a Recycle Bin retention rule, the volume is retained in the Recycle Bin instead of being immediately deleted. For more information, see Recycle Bin.

Console

To delete an EBS volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume. Verify that the volume is in the **Available** state.
4. Choose **Actions**, **Delete volume**.

   If this option is disabled, the volume is attached to an instance and can't be deleted.
5. When prompted for confirmation, enter **`delete`**, and then choose **Delete**.

AWS CLI

To check whether an EBS volume is in use

Use the describe-volumes command. If the volume is in use, the state is `in-use`. Otherwise, it is `available`.

```
aws ec2 describe-volumes \
    --volume-id vol-01234567890abcdef \
    --query Volumes[*].State \
    --output text
```

To delete an EBS volume

Use the delete-volume command.

```
aws ec2 delete-volume --volume-id vol-01234567890abcdef
```

PowerShell

To check whether an EBS volume is in use

Use the Get-EC2Volume cmdlet. If the volume is in use, the state is `in-use`. Otherwise, it is `available`.

```
(Get-EC2Volume `
    -VolumeId vol-01234567890abcdef).State.Value
```

To delete an EBS volume

Use the Remove-EC2Volume cmdlet.

```
Remove-EC2Volume -VolumeId vol-01234567890abcdef
```

Replace an Amazon EBS volume using a snapshot

Amazon EBS snapshots are the preferred backup tool on Amazon EC2 because of their speed, convenience, and cost. When creating a volume from a snapshot, you recreate its state at a specific point in time with the data saved up to that specific point intact. By attaching a volume created from a snapshot to an instance, you can duplicate data across Regions, create test environments, replace a damaged or corrupted production volume in its entirety, or retrieve specific files and directories and transfer them to another attached volume. For more information, see Amazon EBS snapshots.

You can use one of the following procedures to replace an Amazon EBS volume with another volume created from a previous snapshot of that volume.

Requirement

You must create the volume in the same Availability Zone as the instance. Volumes must be attached to instances in the same Availability Zone.

Console

To replace a volume

1. Create a volume from the snapshot and write down the ID of the new volume. For more information, see Create an Amazon EBS volume.
2. On the Instances page, select the instance on which to replace the volume and write down the instance ID.

   With the instance still selected, choose the **Storage** tab. In the **Block devices** section, find the volume to replace and write down the device name for the volume, for example `/dev/sda1`.
3. On the **Storage** tab, choose the volume ID, and then unmount and detach the volume from the instance.
4. Select the new volume that you created in step 1 and choose **Actions**, **Attach volume**.

   For **Instance** and **Device name**, enter the instance ID and device name that you wrote down in Step 2, and then choose **Attach volume**.
5. Connect to your instance and mount the volume. For more information, see Make an Amazon EBS volume available for use.

AWS CLI

To replace a volume

1. Create a new volume from the snapshot. Use the create-volume command with the `--snapshot-id` option. For `--availability-zone`, specify the same Availability Zone as the instance. Note the ID of the new volume in the output.

```
aws ec2 create-volume \
    --volume-type gp3 \
    --snapshot-id snap-0abcdef1234567890 \
    --availability-zone us-east-1a
```

2. Get the device name of the volume to replace. Use the describe-instances command. For `--instance-ids`, specify the ID of the instance on which to replace the volume. Note the device name and volume ID of the volume to replace.

```
aws ec2 describe-instances \
    --instance-ids i-1234567890abcdef0 \
    --query Reservations[].Instances[].BlockDeviceMappings
```

3. Detach the volume to replace from the instance. Use the detach-volume command.

```
aws ec2 detach-volume --volume-id vol-xxxxxxxxxxxxxxxxx
```

4. Attach the replacement volume to the instance. Use the attach-volume command. For `--volume-id`, specify the ID of the replacement volume. For `--instance-id`, specify the ID of the instance on which to attach the volume. For `--device`, specify the same device name that you noted previously.

```
aws ec2 attach-volume \
    --volume-id vol-01234567890abcdef \
    --instance-id i-1234567890abcdef0 \
    --device /dev/sdf
```

5. Connect to your instance and mount the volume. For more information, see Make an Amazon EBS volume available for use.

PowerShell

To replace a volume

1. Create a new volume from the snapshot. Use the New-EC2Volume cmdlet with the `-SnapshotId` option. For `-AvailabilityZone`, specify the same Availability Zone as the instance. Note the ID of the new volume in the output.

```
New-EC2Volume `
    -VolumeType gp3 `
    -SnapshotId snap-0abcdef1234567890 `
    -AvailabilityZone us-east-1a
```

2. Get the device name of the volume to replace. Use the Get-EC2Instance cmdlet. For `-InstanceId`, specify the ID of the instance on which to replace the volume. Note the device name and volume ID of the volume to replace.

```
(Get-EC2Instance `
```

```
    -InstanceId i-1234567890abcdef0).Instances.BlockDeviceMappings | `
     Format-Table DeviceName, @{Name="VolumeId";Expression={$_.Ebs.VolumeId}}
```

3. Detach the volume to replace from the instance. Use the Dismount-EC2Volume cmdlet.

```
DismountEC2Volume -VolumeId vol-xxxxxxxxxxxxxxxxx
```

4. Attach the replacement volume to the instance. Use the Add-EC2Volume cmdlet. For `-VolumeId`, specify the ID of the replacement volume. For `-InstanceId`, specify the ID of the instance on which to attach the volume. For `-Device`, specify the same device name that you noted previously.

```
Add-EC2Volume`
    -VolumeId vol-01234567890abcdef `
    -InstanceId i-1234567890abcdef0 `
    -Device /dev/sdf
```

5. Connect to your instance and mount the volume. For more information, see Make an Amazon EBS volume available for use.

Amazon EBS volume status checks

Volume status checks enable you to better understand, track, and manage potential inconsistencies in the data on an Amazon EBS volume. They are designed to provide you with the information that you need to determine whether your Amazon EBS volumes are impaired, and to help you control how a potentially inconsistent volume is handled.

Volume status checks are automated tests that run every 5 minutes and return a pass or fail status. If all checks pass, the status of the volume is `ok`. If a check fails, the status of the volume is `impaired`. If the status is `insufficient-data`, the checks may still be in progress on the volume. You can view the results of volume status checks to identify any impaired volumes and take any necessary actions.

When Amazon EBS determines that a volume's data is potentially inconsistent, the default is that it disables I/O to the volume from any attached EC2 instances, which helps to prevent data corruption. After I/O is disabled, the next volume status check fails, and the volume status is `impaired`. In addition, you'll see an event that lets you know that I/O is disabled, and that you can resolve the impaired status of the volume by enabling I/O to the volume. We wait until you enable I/O to give you the opportunity to decide whether to continue to let your instances use the volume, or to run a consistency check using a command, such as **fsck** (Linux instances) or **chkdsk** (Windows instances), before doing so.

> **Note**
>
> Volume status is based on the volume status checks, and does not reflect the volume state. Therefore, volume status does not indicate volumes in the `error` state (for example, when a volume is incapable of accepting I/O.) For information about volume states, see Volume states.

If the consistency of a particular volume is not a concern, and you'd prefer that the volume be made available immediately if it's impaired, you can override the default behavior by configuring the volume to automatically enable I/O. If you enable the **Auto-Enable IO** volume attribute (`autoEnableIO` in the API), the volume status check continues to pass. In addition, you'll see an event that lets you know that the volume was determined to be potentially inconsistent, but that its I/O was automatically enabled. This enables you to check the volume's consistency or replace it at a later time.

The I/O performance status check compares actual volume performance to the expected performance of a volume. It alerts you if the volume is performing below expectations. This status check is available only for Provisioned IOPS SSD (`io1` and `io2`) and General Purpose SSD (`gp3`) volumes that are attached to an instance. The status check is not valid for General Purpose SSD (`gp2`), Throughput Optimized HDD (`st1`), Cold HDD (`sc1`), or Magnetic(`standard`) volumes. The I/O performance status check is performed once every minute, and CloudWatch collects this data every 5 minutes. It might take up to 5 minutes from the moment that you attach an `io1` or `io2` volume to an instance for the status check to report the I/O performance status.

> **Important**
>
> While initializing Provisioned IOPS SSD volumes that were restored from snapshots, the performance of the volume may drop below 50 percent of its expected level, which causes the volume to display a `warning` state in the **I/O Performance** status check. This is expected, and you can ignore the `warning` state on Provisioned IOPS SSD volumes while you are initializing them. For more information, see Manually initialize the volumes after creation.

The following table lists statuses for Amazon EBS volumes.

Volume status	I/O enabled status	I/O performance status (io1, io2, and gp3 volumes only)
`ok`	Enabled (I/O Enabled or I/O Auto-Enabled)	Normal (Volume performance is as expected)
`warning`	Enabled (I/O Enabled or I/O Auto-Enabled)	Degraded (Volume performance is below expectations) Severely Degraded (Volume performance is well below expectations)
`impaired`	Enabled (I/O Enabled or I/O Auto-Enabled) Disabled (Volume is offline and pending recovery, or is waiting for the user to enable I/O)	Stalled (Volume performance is severely impacted) Not Available (Unable to determine I/O performance because I/O is disabled)
`insufficient-data`	Enabled (I/O Enabled or I/O Auto-Enabled) Insufficient Data	Insufficient Data

Console

To view status checks

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.

   The **Volume status** column displays the operational status of each volume.

3. To view the status details of a specific volume, select it in the grid and choose the **Status checks** tab.

4. If you have a volume with a failed status check (status is `impaired`), see Work with an impaired Amazon EBS volume.

Alternatively, you can choose **Events** in the navigator to view all the events for your instances and volumes. For more information, see Amazon EBS volume events.

AWS CLI

To view volume status information

Use the describe-volume-status command.

```
aws ec2 describe-volume-status --volume-ids vol-01234567890abcdef
```

Use the following example to identify impaired volumes.

```
aws ec2 describe-volume-status --filters Name=volume-status.status,Values=impaired
```

PowerShell

To view volume status information

Use the Get-EC2VolumeStatus cmdlet.

```
Get-EC2VolumeStatus -VolumeId vol-01234567890abcdef
```

Use the following example to identify impaired volumes.

```
Get-EC2VolumeStatus -Filter @{Name="volume-status.status"; Values="impaired"}
```

Amazon EBS volume events

When Amazon EBS determines that a volume's data is potentially inconsistent, it disables I/O to the volume from any attached EC2 instances by default. This causes the volume status check to fail, and creates a volume status event that indicates the cause of the failure.

To automatically enable I/O on a volume with potential data inconsistencies, change the setting of the **Auto-Enabled IO** volume attribute (`autoEnableIO` in the API). For more information about changing this attribute, see Work with an impaired Amazon EBS volume.

Each event includes a start time that indicates the time at which the event occurred, and a duration that indicates how long I/O for the volume was disabled. The end time is added to the event when I/O for the volume is enabled.

Volume status events

`Awaiting Action: Enable IO`

Volume data is potentially inconsistent. I/O is disabled for the volume until you explicitly enable it. The event description changes to **IO Enabled** after you explicitly enable I/O.

`IO Enabled`

I/O operations were explicitly enabled for this volume.

`IO Auto-Enabled`

I/O operations were automatically enabled on this volume after an event occurred. We recommend that you check for data inconsistencies before continuing to use the data.

`Normal`

For `io1`, `io2`, and `gp3` volumes only. Volume performance is as expected.

`Degraded`

For `io1`, `io2`, and `gp3` volumes only. Volume performance is below expectations.

`Severely Degraded`

For `io1`, `io2`, and `gp3` volumes only. Volume performance is well below expectations.

`Stalled`

For `io1`, `io2`, and `gp3` volumes only. Volume performance is severely impacted.

If you have a volume where I/O is disabled, see Work with an impaired Amazon EBS volume. If you have a volume where I/O performance is below normal, this might be a temporary condition due to an action you have taken (for example, creating a snapshot of a volume during peak usage, running the volume on an instance that cannot support the I/O bandwidth required, accessing data on the volume for the first time, etc.).

Console

To view events for your volumes

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Events**. All instances and volumes that have events are listed.
3. You can filter by volume to view only volume status. You can also filter on specific status types.
4. Select a volume to view its specific event.

AWS CLI

To view events for your volumes

Use the describe-volume-status command.

```
aws ec2 describe-volume-status --volume-ids vol-01234567890abcdef
```

PowerShell

To view events for your volumes

Use the Get-EC2VolumeStatus cmdlet.

```
Get-EC2VolumeStatus -VolumeId vol-01234567890abcdef
```

Work with an impaired Amazon EBS volume

Use the following options if a volume is impaired because the volume's data is potentially inconsistent.

Options

- Option 1: Perform a consistency check on the volume attached to its instance

- Option 2: Perform a consistency check on the volume using another instance
- Option 3: Delete the volume if you no longer need it

Option 1: Perform a consistency check on the volume attached to its instance

The simplest option is to enable I/O and then perform a data consistency check on the volume while the volume is still attached to its Amazon EC2 instance.

To perform a consistency check on an attached volume

1. Stop any applications from using the volume.
2. Enable I/O on the volume. Use one of the following methods.

   Console

   To enable I/O for a volume

   1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
   2. In the navigation pane, choose **Events**.
   3. Select the volume.
   4. Choose **Actions**, **Enable I/O**.

   AWS CLI

   To enable I/O for a volume

   Use the enable-volume-io command.

   ```
   aws ec2 enable-volume-io --volume-id vol-01234567890abcdef
   ```

   PowerShell

   To enable I/O for a volume

   Use the Enable-EC2VolumeIO cmdlet.

   ```
   Enable-EC2VolumeIO -VolumeId vol-01234567890abcdef
   ```

3. Check the data on the volume.

a. Run the **fsck** (Linux instances) or **chkdsk** (Windows instances) command.

b. (Optional) Review any available application or system logs for relevant error messages.

c. If the volume has been impaired for more than 20 minutes, you can contact the AWS Support Center. Choose **Troubleshoot**, and then in the **Troubleshoot Status Checks** dialog box, choose **Contact Support** to submit a support case.

Option 2: Perform a consistency check on the volume using another instance

Use the following procedure to check the volume outside your production environment.

> **Important**
>
> This procedure may cause the loss of write I/Os that were suspended when volume I/O was disabled.

To perform a consistency check on a volume in isolation

1. Stop any applications from using the volume.
2. Detach the volume from the instance. For more information, see [Detach an Amazon EBS volume from an Amazon EC2 instance](#).
3. Enable I/O on the volume. Use one of the following methods.

   Console

   To enable I/O for a volume

   1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
   2. In the navigation pane, choose **Events**.
   3. Select the volume that you detached in the previous step.
   4. Choose **Actions**, **Enable I/O**.

   AWS CLI

   To enable I/O for a volume

   Use the [enable-volume-io](#) command.

```
aws ec2 enable-volume-io --volume-id vol-01234567890abcdef
```

PowerShell

To enable I/O for a volume

Use the Enable-EC2VolumeIO cmdlet.

```
Enable-EC2VolumeIO -VolumeId vol-01234567890abcdef
```

4. Attach the volume to another instance. For more information, see Launch your instance and Attach an Amazon EBS volume to an Amazon EC2 instance.
5. Check the data on the volume.

   a. Run the **fsck** (Linux instances) or **chkdsk** (Windows instances) command.

   b. (Optional) Review any available application or system logs for relevant error messages.

   c. If the volume has been impaired for more than 20 minutes, you can contact the AWS Support Center. Choose **Troubleshoot**, and then in the troubleshooting dialog box, choose **Contact Support** to submit a support case.

Option 3: Delete the volume if you no longer need it

If you want to remove the volume from your environment, simply delete it. For information about deleting a volume, see Delete an Amazon EBS volume.

If you have a recent snapshot that backs up the data on the volume, you can create a new volume from the snapshot. For more information, see Create an Amazon EBS volume.

Auto-enable I/O for impaired Amazon EBS volumes

When Amazon EBS determines that a volume's data is potentially inconsistent, it disables I/O to the volume from any attached EC2 instances by default. This causes the volume status check to fail, and creates a volume status event that indicates the cause of the failure. If the consistency of a particular volume is not a concern, and you prefer that the volume be made available immediately if it's **impaired**, you can override the default behavior by configuring the volume to automatically enable I/O. If you enable the **Auto-Enabled IO** volume attribute (`autoEnableIO` in the API), I/O between the volume and the instance is automatically re-enabled and the volume's status check will pass. In addition, you'll see an event that lets you know that the volume was in a potentially inconsistent state, but that its I/O was automatically enabled. When this event occurs, you should check the volume's consistency and replace it if necessary. For more information, see Amazon EBS volume events.

Console

To view the Auto-Enabled IO attribute of a volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume and choose the **Status checks** tab.

   The **Auto-enabled I/O** field displays the current setting (**Enabled** or **Disabled**) for the selected volume.

To modify the Auto-Enabled IO attribute of a volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume and choose **Actions**, **Manage auto-enabled I/O**.
4. To automatically enable I/O for an impaired volume, select the **Auto-enable I/O for impaired volumes** check box. To disable the feature, clear the check box.
5. Choose **Update**.

AWS CLI

To view the autoEnableIO attribute of a volume

Use the describe-volume-attribute command.

```
aws ec2 describe-volume-attribute \
    --attribute autoEnableIO \
    --volume-id vol-01234567890abcdef
```

The following is example output.

```
{
```

```
    "AutoEnableIO": {
        "Value": true
    },
    "VolumeId": "vol-01234567890abcdef"
}
```

To modify the autoEnableIO attribute of a volume

Use the [modify-volume-attribute](#) command.

```
aws ec2 modify-volume-attribute \
    --auto-enable-io \
    --volume-id vol-01234567890abcdef
```

PowerShell

To view the autoEnableIO attribute of a volume

Use the [Get-EC2VolumeAttribute](#) cmdlet.

```
(Get-EC2VolumeAttribute `
    -Attribute autoEnableIO `
    -VolumeId vol-01234567890abcdef).AutoEnableIO
```

The following is example output.

```
True
```

To modify the autoEnableIO attribute of a volume

Use the [Edit-EC2VolumeAttribute](#) cmdlet.

```
Edit-EC2VolumeAttribute `
    -AutoEnableIO $true `
    -VolumeId vol-01234567890abcdef
```

Fault testing on Amazon EBS

AWS Fault Injection Service (AWS FIS) is a fully managed service that helps you perform fault injection experiments on your AWS workloads. With EBS actions in AWS FIS, you can test how your applications respond to storage faults that can result in I/O interruptions and degraded performance on your volumes. This controlled testing environment enables you to observe how your applications respond to disruptions so you can identify weaknesses in your architecture and improve the overall resilience of your applications. Using the pause I/O action and the latency injection action, you can test your monitoring and recovery mechanisms such as Amazon CloudWatch alarms and failover workflows, and improve the resiliency of your mission-critical applications to storage faults. For more information about AWS FIS, see the AWS Fault Injection Service User Guide.

Available experiments

Amazon EBS currently supports two AWS FIS fault injections:

- Pause I/O fault injection
- Latency injection

Considerations

The following considerations apply:

- All Amazon EBS volume types are supported. Both root volumes and data volumes are supported. Instance store volumes are not supported.
- Volumes must be attached to Nitro-based EC2 instances.
- Your volumes will resume their original I/O performance once the experiment completes based on the duration. You can also stop a running experiment before it completes. Alternatively, you can create a stop condition to stop the experiment if it reaches a threshold that you define in a CloudWatch alarm.
- You can use AWS FIS with Multi-Attach enabled volumes. All of the attached instances are impacted. You can't select a specific volume-instance attachment for experiments.
- FIS is currently not available in Local Zones, Outposts, or Wavelength Zones.
- You can test up to 5 volumes in the same Availability Zone simultaneously when specifying volume ARNs in the console.
- You can't use AWS FIS with volumes created on an Outpost, in an AWS Wavelength Zone, or in a Local Zone.

Pause I/O fault injection

Use AWS Fault Injection Service and the Pause I/O action to temporarily stop I/O between an Amazon EBS volume and the instances to which it is attached to test how your workloads handle I/O interruptions.

For more information about AWS FIS, see the *AWS Fault Injection Service User Guide*.

Considerations

Keep in mind the following considerations for pausing volume I/O:

- Pause I/O is supported on all Nitro-based instance types.
- To test your OS timeout configuration, set the experiment duration equal to or greater than the value specified for `nvme_core.io_timeout`. For more information, see NVMe I/O operation timeout for Amazon EBS volumes.
- If you drive I/O to a volume that has I/O paused, the following happens:
  - The volume's status transitions to `impaired` within 120 seconds. For more information, see Amazon EBS volume status checks.
  - The CloudWatch metric for `VolumeStalledIOCheck` will be `1` if volume I/O is paused for over 60 seconds. For more information see Metrics for Amazon EBS volumes.
  - The CloudWatch metrics for queue length (`VolumeQueueLength`) will be non-zero. Any alarms or monitoring should monitor for a non-zero queue depth.
  - The CloudWatch metrics for `VolumeReadOps` or `VolumeWriteOps` will be `0`, which indicates that the volume is no longer processing I/O.

You can perform a basic experiment from the Amazon EC2 console, or you can perform more advanced experiments using the AWS FIS console. For more information about performing advanced experiments using the AWS FIS console, see Tutorials for AWS FIS in the *AWS Fault Injection Service User Guide*.

To perform a basic experiment using the Amazon EC2 console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes.**
3. Select the volume for which to pause I/O and choose **Actions**, **Fault injection**, **Pause volume I/O**.

4. For **Duration**, enter the duration for which to pause I/O between the volume and the instances. The field next to the Duration dropdown list shows the duration in ISO 8601 format.
5. In the **Service access** section, select the IAM service role for AWS FIS to assume to perform the experiment. You can use either the default role, or an existing role that you created. For more information, see Create an IAM role for AWS FIS experiments.
6. Choose **Pause volume I/O**. When prompted, enter `start` in the confirmation field and choose **Start experiment**.
7. Monitor the progress and impact of your experiment. For more information, see Monitoring AWS FIS in the *AWS FIS User Guide*.

Latency injection

Use the Latency Injection action (`aws:ebs:volume-io-latency`) in AWS FIS to simulate elevated I/O latency on your Amazon EBS volumes to test how your applications respond to storage performance degradation. This action allows you to specify the latency value to be injected as well as the percentage of I/O that will be impacted on the target volume. With AWS FIS, you can use pre-configured latency experiment templates to get started with testing different I/O latency patterns that may be observed during storage faults. These templates are designed as an initial set of scenarios you can use to introduce disruptions to your applications to test resiliency. They are not designed to encompass all types of impact your applications can experience in the real world. We recommend that you to adapt them to run multiple different tests based on the performance needs of your applications. You can customize the available templates or create new experiment templates to test for your application specific requirements.

Pre-configured latency experiment templates

Amazon EBS provides the following latency experiment templates through the EBS Console and the AWS FIS scenario library. You can directly use these templates on your target volumes to run a latency injection experiment.

- **Sustained Latency** — Simulates constant latency. This experiment utilizes one latency injection action and has a total duration of 15 minutes. This experiment simulates persistent latency on 50 percent of read I/O and 100 percent of write I/O: 500 ms for 15 minutes.
- **Increasing Latency** — Simulates gradually increasing latency. This experiment utilizes five latency injection actions and has a total duration of 15 minutes. This experiment will simulate a gradual increase in latency on 10 percent of read I/O and 25 percent of write I/O: 50 ms for 3

minutes, 200 ms for 3 minutes, 700 ms for 3 minutes, 1 second for 3 minutes, and 15 seconds for 3 minutes.

- **Intermittent Latency** — Simulates sharp intermittent latency spikes with periods of recovery in between. This experiment utilizes three latency injection actions and has a total duration of 15 minutes. This experiment will simulate three latency spikes on 0.1 percent of read and write I/O: 30 second spike that lasts for 1 minute, 10 second spike that lasts for 2 minutes, and 20 second spike that lasts for 2 minutes. There will be 5 minute periods of recovery between each latency spike.
- **Decreasing Latency** — Simulates gradually decreasing latency. This experiment utilizes five latency injection actions and has a total duration of 15 minutes. This experiment will simulate a gradual decrease in latency on 10 percent of read I/O and write I/O: 20 seconds for 3 minutes, 5 seconds for 3 minutes, 900 ms for 3 minutes, 300 ms for 3 minutes, and 40 ms for 3 minutes.

Customize preconfigured scenarios

You customize the preconfigured templates above or create your own new experiment templates using the following customizable parameters.

- `readIOPercentage` — Percentage of read I/O operations that latency will be injected on. This is the percentage of all read I/O operations on the volume that will be impacted by the action.

  Range: Min 0.1%, Max 100%
- `readIOLatencyMilliseconds` — Amount of latency injected on read I/O operations. This is the latency value that will be observed on the specified percentage of the read I/O during the experiment.

  Range: Min 1 ms (io2) / 10 ms (non-io2), Max 60 seconds
- `writeIOPercentage` — Percentage of write I/O operations that latency will be injected on. This is the percentage of all write I/O operations on the volume that will be impacted by the action.

  Range: Min 0.1%, Max 100%
- `writeIOLatencyMilliseconds` — Amount of latency injected on write I/O operations. This is the latency value that will be observed on the specified percentage of the write I/O during the experiment.

  Range: Min 1ms (io2) / 10ms (non-io2), Max 60 seconds

- `duration` — Duration for which the latency will be injected on the percentage of I/O selected.

  Range: Min 1 second, Max 12 hours

Monitoring latency injection

You can monitor the performance impact on your volumes in the following ways:

- Use average latency metrics in CloudWatch to get per-minute average I/O latency. For more information, see Monitor your EBS volumes using CloudWatch.
- Use EBS detailed performance statistics available through NVMe-CLI, CloudWatch agent, and Prometheus to get per-second average I/O latency. The detailed metrics also provide I/O latency histograms that you can use to analyze latency variance on your volumes. For more information, see NVMe detailed performance statistics.
- Use the Amazon EBS volume status checks. When you inject I/O latency, the volume's status transitions to the `warning` state.

Considerations

Consider the following when using EBS latency injection:

- Latency injection is supported on all Nitro-based instance types, except: P4d, P5, P5e, Trn2u, G6, G6f, Gr6, Gr6f, M8i, M8i-flex, C8i-flex, R8i, R8i-flex, I8ge, Mac-m4pro, and Mac-m4.
- You might see up to 5 percent variance in the latency value specified in the experiment and the resultant latency observed.
- If you drive a very small number of I/O operations, the percentage of I/O specified in the action parameters might not match the actual percentage of I/O impacted by the action.

To run a latency injection experiment on an Amazon EBS volume

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volumes on which to run the experiment and choose **Actions**, **Resilience testing**, **Inject volume I/O latency**.

   The AWS Fault Injection Service console opens.

4. In the **Create experiment** window, select the type of experiment to run: **Intermittent**, **Increasing**, **Sustained**, or **Decreasing**.
5. For **IAM role selection**, choose **Create a new role** to create a new role that AWS FIS will use to conduct the experiments on your behalf. Alternatively, choose **Use an existing IAM role** if you previously created an IAM role with the required permissions.
6. The **Pricing estimate** section gives you an estimate of the cost of running the experiment. With AWS FIS, you are charged per minute that an action runs, from start to finish, based on the number of target accounts for your experiment.
7. Choose **Start experiment**.

Amazon EBS snapshots

You can back up the data on your Amazon EBS volumes by making point-in-time copies, known as *Amazon EBS snapshots*. A snapshot is an **incremental backup**, which means that we save only the blocks on the volume that have changed since the most recent snapshot. This minimizes the time required to create the snapshot and saves on storage costs by not duplicating data.

> **Important**
>
> AWS does not automatically back up the data stored on your EBS volumes. For data resiliency and disaster recovery, it is your responsibility to create EBS snapshots on a regular basis, or to set up automatic snapshot creation by using Automate backups with Amazon Data Lifecycle Manager or AWS Backup.

Snapshots are stored in Amazon S3, in S3 buckets that you can't access directly. You can create and manage your snapshots using the Amazon EC2 console or the Amazon EC2 API. You can't access your snapshots using the Amazon S3 console or the Amazon S3 API.

Snapshot data is automatically replicated across all Availability Zones in the Region. This provides high availability and durability for snapshot data, and enables you to restore volumes in any Availability Zones in that Region.

Each snapshot contains all of the information that is needed to restore your data (from the moment when the snapshot was taken) to a new EBS volume. When you create an EBS volume from a snapshot, the new volume begins as an exact replica of the volume that was used to create the snapshot.

For more information, see the Amazon EBS Snapshots product page.

Snapshot events

You can track the status of your EBS snapshots through CloudWatch Events. For more information, see EBS snapshot events.

Snapshot pricing

Charges for your snapshots are based on the amount of data stored. Because snapshots are incremental, deleting a snapshot might not reduce your data storage costs. Data referenced exclusively by a snapshot is removed when that snapshot is deleted, but data referenced by other snapshots is preserved. For more information, see [Amazon Elastic Block Store Volumes and Snapshots] in the *AWS Billing User Guide*.

Contents

How Amazon EBS snapshots work

The first snapshot that you create from a volume is always a *full snapshot*. It includes all of the data blocks written to the volume at the time of creating the snapshot. Subsequent snapshots of the same volume are *incremental snapshots*. They include only changed and new data blocks written to the volume since the last snapshot was created

The size of a full snapshot is determined by the size of the data being backed up, not the size of the source volume. Similarly, the storage costs associated with a full snapshot is determined by the size of the snapshot, not the size of the source volume. For example, you create the first snapshot of a `200 GiB` Amazon EBS volume that contains only `50 GiB` of data. This results in a full snapshot that is `50 GiB` in size, and you are billed for `50 GiB` snapshot storage.

Similarly, the size and storage costs of an incremental snapshot are determined by the size of any data that was written to the volume since the previous snapshot was created. Continuing the previous example, if you create a second snapshot of the same `200 GiB` volume after changing `20 GiB` of data and adding `10 GiB` of data, the incremental snapshot is `30 GiB` in size. You are then billed for that additional `30 GiB` snapshot storage.

For more information about snapshot pricing, see [Amazon EBS pricing].

> **Important**
>
> When you archive an incremental snapshot, it is converted to a full snapshot that includes all of the blocks written to the volume at the time that the snapshot was created. It is then

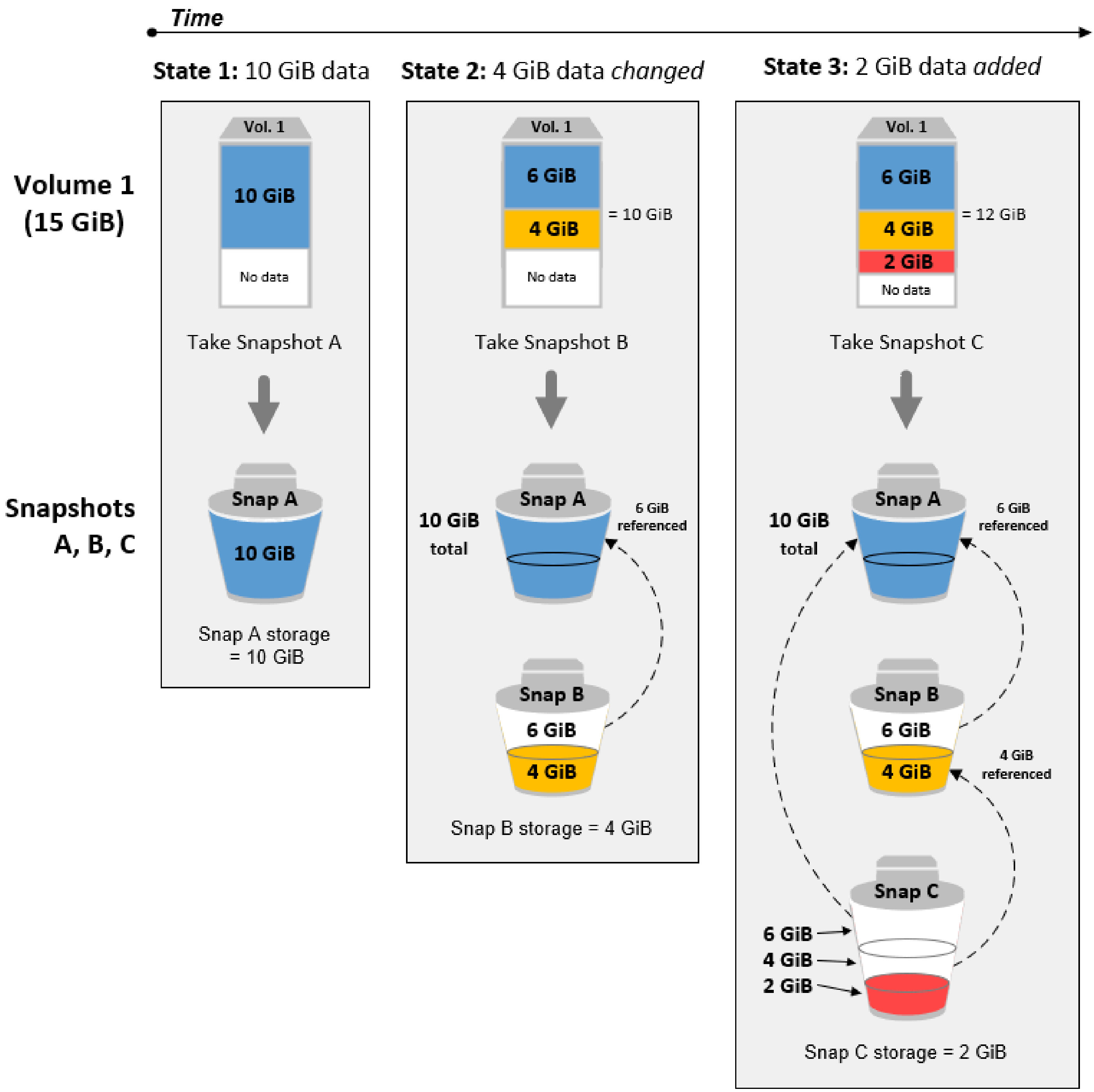
moved to the Amazon EBS Snapshots Archive tier. Snapshots in the archive tier are billed at a different rate from snapshots in the standard tier. For more information, see Pricing and billing for archiving Amazon EBS snapshots.

The following sections show how an EBS snapshot captures the state of a volume at a point in time, and how subsequent snapshots of a changing volume create a history of those changes.

Multiple snapshots of the same volume

The diagram in this section shows Volume 1, which is `15 GiB` in size, at three points in time. A snapshot is taken of each of these three volume states. The diagram specifically shows the following:

- In **State 1**, the volume has `10 GiB` of data. **Snap A** is the first snapshot taken of the volume. **Snap A** is a full snapshot and the entire `10 GiB` of data is backed up.
- In **State 2**, the volume still contains `10 GiB` of data, but only `4 GiB` have changed after **Snap A** was taken. **Snap B** is an incremental snapshot. It needs to back up only the `4 GiB` that changed. The other `6 GiB` of unchanged data, which are already backed up in **Snap A**, are *referenced* by **Snap B** rather than being backed up again. This is indicated by the dashed arrow.
- In **State 3**, `2 GiB` of data have been added to the volume, for a total of `12 GiB`, after **Snap B** was taken. **Snap C** is an incremental snapshot. It needs to back up only the `2 GiB` that were added after **Snap B** was taken. As shown by the dashed arrows, **Snap C** also references the `4 GiB` of data stored in **Snap B**, and the `6 GiB` of data stored in **Snap A**.
- The total storage required for the three snapshots is `16 GiB` total. This accounts for 10 GiB for Snap A, 4 GiB for Snap B, and 2 GiB for Snap C.



Incremental snapshots of different volumes

The diagram in this section shows how incremental snapshots can be taken from different volumes.

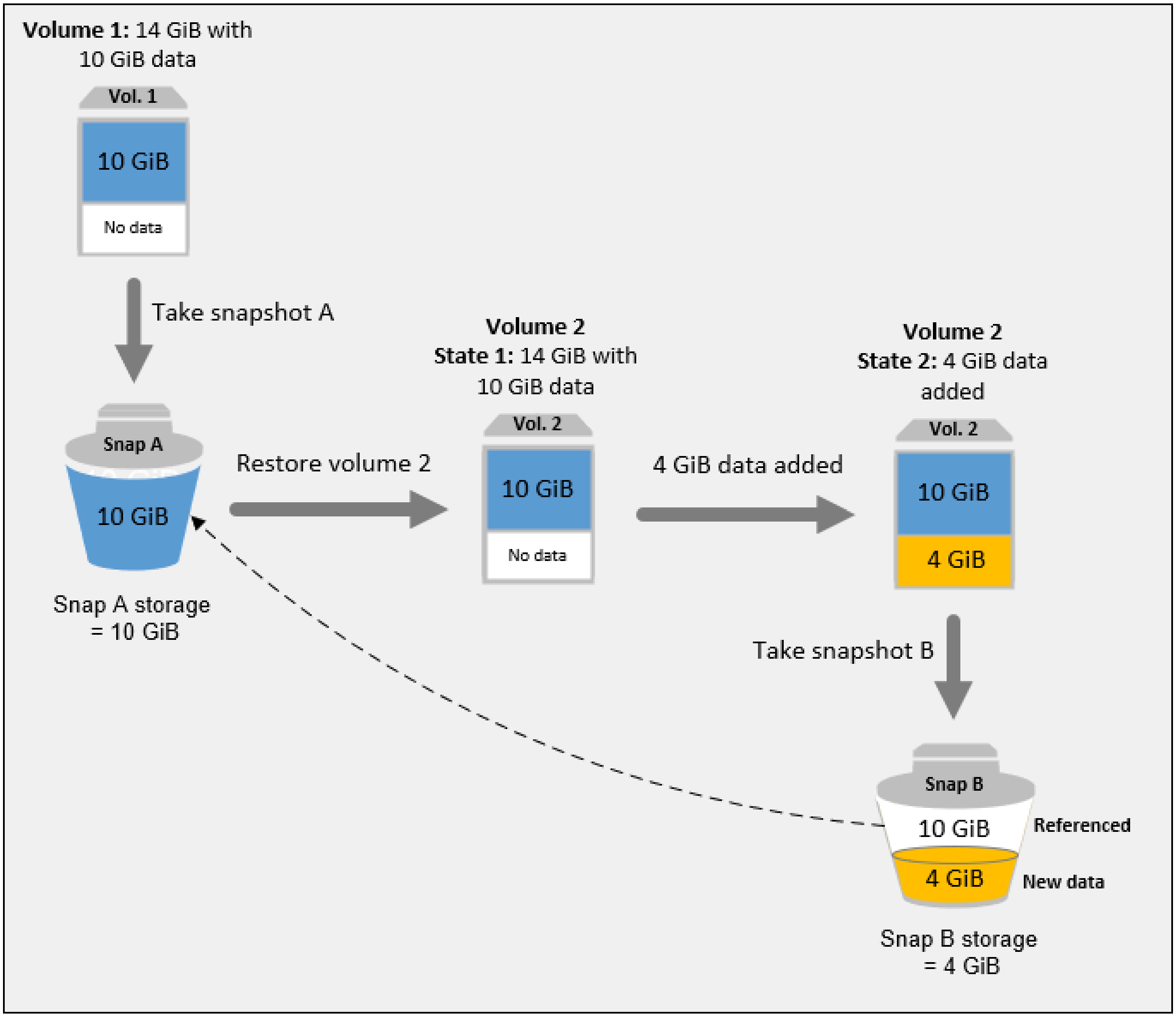
1. **Vol 1**, which is `14 GiB` in size, has `10 GiB` of data. Because **Snap A** is the first snapshot taken of the volume, it is a full snapshot and the entire `10 GiB` of data is backed up.

2. **Vol 2** is created from **Snap A**, so it is an exact replica of **Vol 1** at the time the snapshot was taken.
3. Over time, `4 GiB` of data is added to **Vol 2** and the total size of its data is `14 GiB`.
4. **Snap B** is taken from **Vol 2**. For **Snap B**, only the `4 GiB` of data that was added after the volume was created from **Snap A** is backed up. The other `10 GiB` of unchanged data, which is already stored in **Snap A**, is referenced by **Snap B** instead of being backed up again.

   Snap B is an incremental snapshot of **Snap A**, even though it was created from a different volume.

Important

The diagram assumes that you own **Vol 1** and **Snap A**, and that **Vol 2** is encrypted with the same KMS key as Vol 1. If **Vol 1** was owned by another AWS account and that account took **Snap A** and shared it with you, then **Snap B** would be a full snapshot. Or, if **Vol 2** was encrypted with a different KMS key than **Vol 1**, then **Snap B** would be a full snapshot.

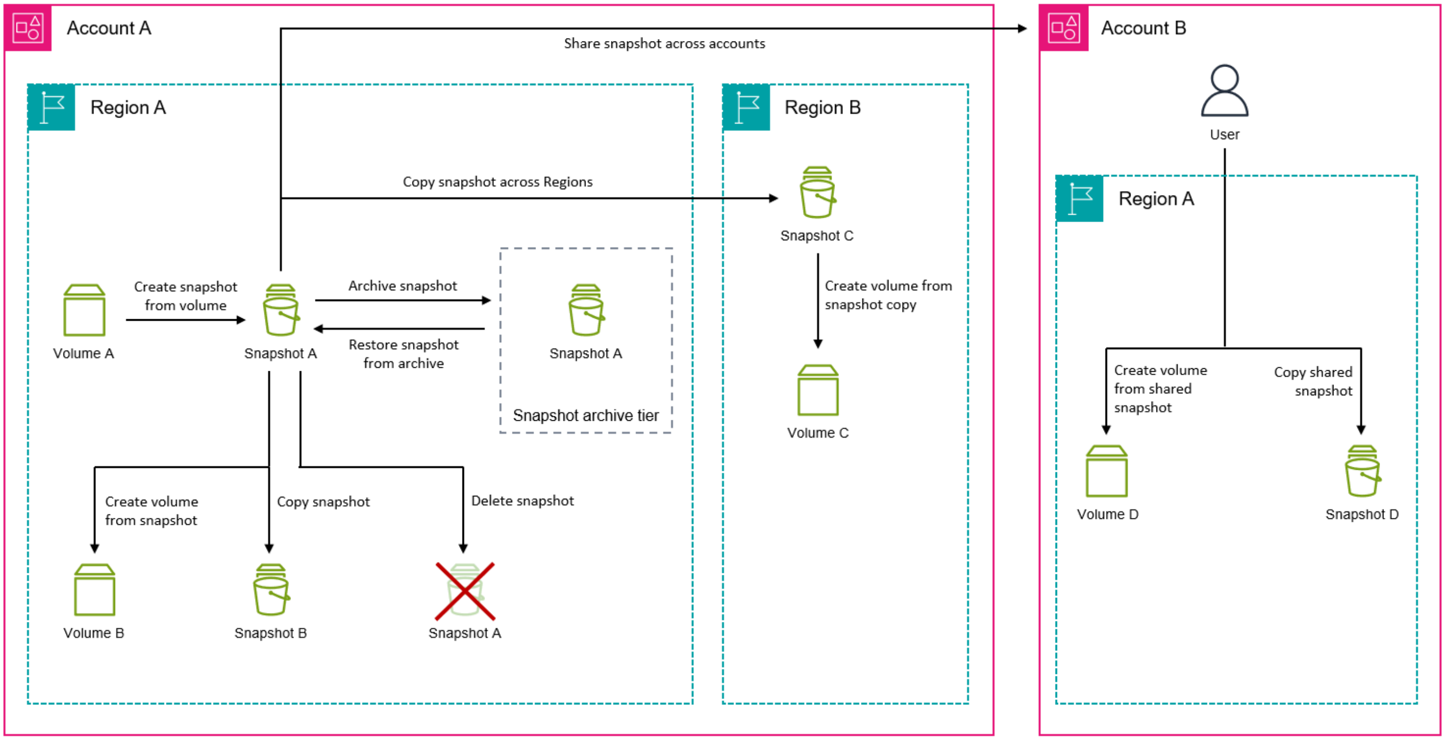


For more information about how data is managed when you delete a snapshot, see Delete an Amazon EBS snapshot.

Amazon EBS snapshot lifecycle

The lifecycle of an Amazon EBS snapshot starts with the creation process. You create snapshots from Amazon EBS volumes. You can use snapshots to restore new Amazon EBS volumes. You can create copies of snapshots either in the same Region, or in different Regions. You can share snapshots with other AWS accounts, either publicly or privately. Those accounts can restore volumes from the shared snapshots, or they can create copies of the shared snapshots in their own account. If you don't need immediate access to a snapshot, you can archive it to save on storage costs.

The following image shows actions that you can perform on your snapshots as part of the snapshot lifecycle.



Tasks

- Create Amazon EBS snapshots
- View Amazon EBS snapshot information
- Copy an Amazon EBS snapshot
- Share an Amazon EBS snapshot with other AWS accounts
- Archive Amazon EBS snapshots
- Delete an Amazon EBS snapshot

Create Amazon EBS snapshots

You can create an Amazon EBS snapshot of an Amazon EBS volume to create a point-in-time backup of that volume. You can either create snapshots of **individual Amazon EBS volumes**, or you can create **multi-volume snapshots** of all, or a subset, of the volumes attached to an Amazon EC2 instance.

Snapshot creation is asynchronous. The snapshot is created immediately, but it remains in the `pending` state until all of data has been transferred to Amazon S3. This can take several hours to complete, depending on the number of modified blocks on the volume. You can continue to use the volume during this time without impacting the snapshot. The snapshot includes only the data that was written to the volume at the time the snapshot was requested. It does not include data that has been cached by applications or the operating system.

> **Tip**
>
> To ensure consistent and complete snapshots, we recommend that you pause writes to the volume before you create the snapshot. If you can't pause writes to the volume, we recommend that you unmount the volume, from within the instance, before you create the snapshot. You can remount and resume writes once the snapshot enters the `pending` state.
> If you create a snapshot of a volume that serves as the root device for an Amazon EC2 instance, we recommend that you stop the instance before taking the snapshot.

Contents

Snapshot encryption

A snapshot automatically gets the same encryption status as the volume from which it is created. Snapshots created from unencrypted volumes are not encrypted. Snapshots created from encrypted volumes are automatically encrypted using the same KMS key as the volume.

> **Tip**
>
> If you need to create an encrypted snapshot from an unencrypted volume, first create the unencrypted snapshot of the volume, and then create an encrypted copy of that snapshot.

Snapshot destinations

The location of the source resource (volume or instance) determines where you can create snapshots.

- If the source resource is in a Region, you must create snapshots in the same Region as the source resource.
- If the source resource is in a Local Zone, you can create snapshots in the same Local Zone or in its parent Region. For more information, see Local snapshots in Local Zones.
- If the source resource is on an Outpost, you can create snapshots on the same Outpost or in its parent Region. For more information, see Amazon EBS local snapshots on Outposts.

Automating snapshots

You can automate snapshot creation using Amazon Data Lifecycle Manager and AWS Backup.

Considerations for creating snapshots

- We recommend that you do not create snapshots of volumes that are attached to Amazon EC2 instances that are hibernated or that are enabled for hibernation. For more information, see How Amazon EC2 instance hibernation works.
- Although you can take a snapshot of a volume while a previous snapshot of that volume is in the `pending` status, having multiple snapshots in the `pending` state for the same volume can result in reduced volume performance until the snapshots complete.
- There are limits on the number of snapshots you can have in the `pending` state, and on the number of concurrent snapshots you can request per volume type. For more information, see Quotas for Amazon EBS. If you exceed one of these quotas, wait for the current snapshots to complete and then try again.

Create a snapshot of an EBS volume

You can create a single snapshot of a single volume. Alternatively, to automate snapshot creation, use Amazon Data Lifecycle Manager or AWS Backup. To create snapshots of all the volumes for an EC2 instance, use multi-volume snapshots.

Console

To create a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**, **Create snapshot**.
3. For **Resource type**, choose **Volume**.
4. For **Volume ID**, select the volume from which to create the snapshot. The **Encryption** field indicates the volume and resulting snapshot's encryption status. It can't be modified.
5. (*Optional*) For **Description**, enter a brief description for the snapshot.
6. If the volume is on an Outpost or in a Local Zone, the **Snapshot destination** field appears. Do one of the following:
   - If the volume is in a Local Zone, choose **Local Zone** to create the snapshot in the same Local Zone, or choose **AWS Region** to create the snapshot in the parent Region of the Local Zone.
   - If the volume is on an Outpost, choose **AWS Outpost**, to create the snapshot on the same Outpost, or choose **AWS Region** to create the snapshot in the parent Region of the Outpost.

   > **Note**
   >
   > If the volume is in a Region, the **Snapshot destination** does not appear. The snapshot is automatically created in the same Region as the volume.

7. (*Optional*) To assign custom tags to the snapshot, in the **Tags** section, choose **Add tag**, and then enter the key-value pair. You can add up to 50 tags.
8. Choose **Create snapshot**.

AWS CLI

To create a snapshot

Use the create-snapshot command.

```
aws ec2 create-snapshot  \
```

```
    --volume-id vol-01234567890abcdef \
    --description "Snapshot of the root volume for i-1234567890abcdef0"
```

PowerShell

To create a snapshot

Use the New-EC2Snapshot cmdlet.

```
New-EC2Snapshot `
    -VolumeId vol-01234567890abcdef `
    -Description "Snapshot of the root volume for i-1234567890abcdef0"
```

Create multi-volume EBS snapshots from an EC2 instance

By default, when you create multi-volume snapshots from an Amazon EC2 instance, Amazon EBS creates snapshots of all the Amazon EBS volumes that are attached to the instance. However, you can choose to exclude the root volume, or specific data volumes if needed.

> **Tip**
>
> We recommend that you tag your multi-volume snapshots so that it's easy to identify and manage them collectively. You can also copy the tags from the source volumes to the corresponding snapshots to set the snapshot metadata, such as access policies, attachment information, and cost allocation, to match the source volume.

Considerations for multi-volume snapshots

- If all of the snapshots complete successfully, a `createSnapshots` CloudWatch event with a result of `succeeded` is sent to your AWS account. If any one snapshot in the multi-volume snapshot set fails, all of the other snapshots enter the `error` state and a `createSnapshots` CloudWatch event with a result of `failed` is sent to your account. For more information, see Create snapshots (createSnapshots).
- Multi-volume snapshots support up to 128 Amazon EBS volumes attached to an instance, including the root volume and up to 127 data volumes.
- Each snapshot in the multi-volume snapshot set is an individual snapshot that can be used in the same way, and that supports the same features, as an individual snapshot.

- You can take application-consistent snapshots of all the Amazon EBS volumes attached to an Amazon EC2 Windows instance using AWS Systems Manager command documents.

Console

To create multi-volume snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**, **Create snapshot**.
3. For **Resource type**, choose **Instance**.
4. For **Description**, enter a brief description for the snapshots. This description is applied to all of the snapshots.
5. If the instance is on an Outpost or in a Local Zone, the **Snapshot destination** field appears. Do one of the following:
   - If the instance is in a Local Zone, choose **Local Zone** to create the snapshots in the same Local Zone, or choose **AWS Region** to create the snapshots in the parent Region of the Local Zone.
   - If the instance is on an Outpost, choose **AWS Outpost**, to create the snapshots on the same Outpost, or choose **AWS Region** to create the snapshots in the parent Region of the Outpost.

   > **Note**
   >
   > If the instance is in a Region, the **Snapshot destination** does not appear. The snapshot is automatically created in the same Region as the instance.
6. (*Optional*) To exclude the instance's root volume, select **Exclude root volume**.
7. (*Optional*) To exclude data volumes, select **Exclude specific data volumes**. The **Attached data volumes** section lists all of the data volumes that are currently attached to the selected instance.

   Select the data volumes to exclude. Only the volumes that remain unselected will be included in the multi-volume snapshot set.
8. (*Optional*) To automatically copy tags from the source volumes to the corresponding snapshots, for **Copy tags from source volume**, select **Copy tags**.

9. (*Optional*) To assign additional custom tags to the snapshots, in the **Tags** section, choose **Add tag**, and then enter the key-value pair. You can add up to 50 tags.
10. Choose **Create snapshot**.

AWS CLI

To create multi-volume snapshots

Use the create-snapshots command.

```
aws ec2 create-snapshots \
    --instance-specification InstanceId=i-1234567890abcdef0 \
    --description "from a multi-volume snapshot of i-1234567890abcdef0"
```

To exclude the root volume, add the following to the `--instance-specification` option.

```
ExcludeBootVolume=true
```

To exclude a data volume, add the following to the `--instance-specification` option.

```
ExcludeDataVolumeIds=vol-01234567890abcdef
```

PowerShell

To create multi-volume snapshots

Use the New-EC2SnapshotBatch cmdlet.

```
New-EC2SnapshotBatch `
    -InstanceSpecification_InstanceId i-1234567890abcdef0 `
    -Description "from a multi-volume snapshot of i-1234567890abcdef0"
```

To exclude the root volume, add the following parameter.

```
-InstanceSpecification_ExcludeBootVolume $true
```

To exclude data volumes, add the following parameter.

```
-InstanceSpecification_ExcludeDataVolumes "vol-01234567890abcdef"
```

View Amazon EBS snapshot information

You can view detailed information about your snapshots.

Console

To view snapshot information

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. To view only your snapshots that you own, in the top-left corner of the screen, choose **Owned by me**. You can also filter the list of snapshots using tags and other snapshot attributes. In the **Filter** field, select the attribute field, and then select or enter the attribute value. For example, to view only encrypted snapshots, select **Encryption**, and then enter `true`.
4. To view more information about a specific snapshot, choose its ID in the list.

> **Note**
>
> The **Full snapshot size** field shows the full size of the snapshot, in bytes. This is **not** the incremental size of the snapshot. Instead, it represents the size of all the blocks that were written to the source volume at the time the snapshot was created.
> The **Volume size** field shows the size of the EBS volume that will created from the snapshot if no other size is specified.

AWS CLI

To view snapshot information

Use the describe-snapshots command.

Example Example 1: Filter based on tags

The following example describes the snapshots with the tag Stack=production.

```
aws ec2 describe-snapshots --filters Name=tag:Stack,Values=production
```

Example Example 2: Filter based on volume

The following example describes the snapshots created from the specified volume.

```
aws ec2 describe-snapshots --filters Name=volume-id,Values=vol-049df61146c4d7901
```

Example Example 3: Filter based on snapshot age

You can use JMESPath to filter results using expressions. For example, the following command displays the IDs of all snapshots created by your account before the specified date. If you do not specify the owner, the results include all public snapshots.

```
aws ec2 describe-snapshots \
    --filters Name=owner-id,Values=123456789012 \
    --query "Snapshots[?(StartTime<='2024-03-31')].[SnapshotId]" \
    --output text
```

The following command displays the IDs of all snapshots created in the specified date range.

```
aws ec2 describe-snapshots \
    --filters Name=owner-id,Values=123456789012 \
    --query "Snapshots[?(StartTime>='2024-01-01') && (StartTime<='2024-12-31')].
[SnapshotId]" \
    --output text
```

PowerShell

To view snapshot information

Use the Get-EC2Snapshot cmdlet.

Example 1: Describe a snapshot

The following example describes the specified snapshot.

```
Get-EC2Snapshot -SnapshotId snap-0abcdef1234567890
```

Example 2: Filter based on volume

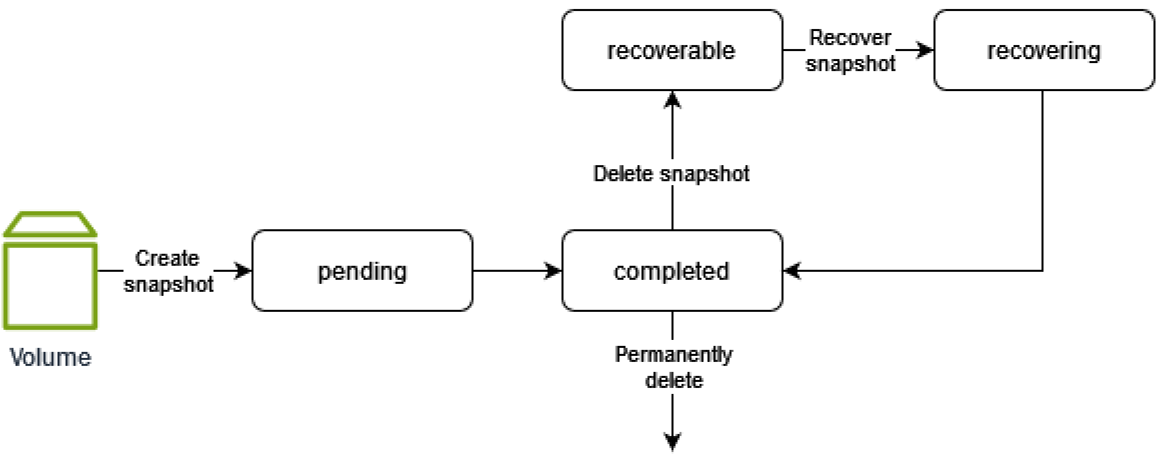
The following example describes the snapshots created from the specified volume.

```
Get-EC2Snapshot`
    -Filter @{Name="volume-id"; Values="vol-01234567890abcdef"}
```

Snapshot states

An Amazon EBS snapshot transitions through different states from the moment it is created until it is permanently deleted.

The following illustration shows the transitions between snapshot states. When you create a snapshot, it enters the `pending` state. After the snapshot is ready for use, it enters the `completed` state. When you've decided that you no longer need a snapshot, you can delete it. If you delete a snapshot that matches a Recycle Bin retention rule, it is retained in the Recycle Bin and it enters the `recoverable` state. If you recover a snapshot from the Recycle Bin, it enters the `recovering` state and then the `completed` state. Otherwise, it is permanently deleted.



The following table summarizes the snapshot states.

Status	Description
`pending`	The snapshot creation process is still in progress. A snapshot can't be used while it is in the `pending` state.
`completed`	The snapshot creation process has completed and the snapshot is ready for use.

Status	Description
`recoverable`	The snapshot is currently in the Recycle Bin. To use the snapshot, you must first recover it from the Recycle Bin.
`recovering`	The snapshot is being recovered from the Recycle Bin. After the snapshot has been recovered, it transitions to the `completed` state and becomes ready for use.
`error`	The snapshot creation process has failed. A snapshot can't be used if it is in the `error` state.

Copy an Amazon EBS snapshot

After you create a snapshot, and it has reached the `completed` state, you can create a copy of it. The snapshot copy is an exact copy of the original, but it has a unique resource ID. You can copy snapshots that you own and snapshots that are shared with you, privately or publicly. You might need to copy a snapshot for the following use cases:

- Geographic expansion — You need to launch your applications in a new Region.
- Migration — You need to move an application to a new destination, to enable better availability or to minimize cost.
- Disaster recovery — You need to back up your data and logs to secondary Regions for data redundancy purposes.
- Encryption — You need to encrypt a previously unencrypted snapshot or reencrypt an encrypted snapshot using a different KMS key.
- Copy a shared snapshot — You need to copy a snapshot that is shared with you.
- Data retention and auditing requirements — You need to copy encrypted snapshots from one AWS account to another to preserve data for auditing or data retention. Using a different account protects you if your main AWS account is compromised.

To copy multi-volume snapshots, identify all of the snapshots that are part of that set using the tags that you assigned during creation, and then copy them individually.

For information about copying an Amazon RDS snapshot, see Copying a DB Snapshot in the *Amazon RDS User Guide*.

Pricing

For pricing information about copying snapshots, see Amazon EBS Pricing.

Contents

Considerations for copying snapshots

- You can copy AWS Marketplace, VM Import/Export, and Storage Gateway snapshots, but you must verify that the snapshot is supported in the destination Region.
- There is a limit of `20` concurrent snapshot copy requests per destination. If you exceed this quota, you receive a `ResourceLimitExceeded` error. If you receive this error, wait for one or more of the copy requests to complete before making a new snapshot copy request.
- User-defined tags are not copied from the source snapshot to the snapshot copy. You can add user-defined tags during or after the copy operation.
- Snapshots created by a snapshot copy operation have an arbitrary volume ID, such as `vol-ffff` or `vol-ffffffff`. These arbitrary volume IDs should not be used for any purpose.
- Resource-level permissions specified for the snapshot copy operation can apply to the snapshot copy and the source snapshot. For an example, see Example: Copying snapshots.
- If you copy a snapshot that is enabled for fast snapshot restore, the snapshot copy is not automatically enabled for fast snapshot restore. You must explicitly enable fast snapshot restore for the snapshot copy.
- If you copy a snapshot and encrypt it to a new KMS key, a complete (non-incremental) copy is created. This results in additional storage costs.

- If you copy a snapshot to a new Region, a full (non-incremental) copy is created. This results in additional storage costs.
- If you use external or cross-Region data transfers, additional EC2 data transfer charges will apply. If you delete any snapshots after initiation, you are still charged for the data that has already been transferred.

Destinations for snapshot copies

The location of the source snapshot determines whether you can copy it or not.

- If the source snapshot is in a Region, you can copy it within that Region, to another Region, to an Outpost associated with that Region, or to a Local Zone in that Region.
- If the source snapshot is in a Local Zone, you can copy it within the same Local Zone, to another Local Zone in the same zone group, or to the parent Region of that Local Zone.
- If the source snapshot is on an Outpost, you can't copy it.

Incremental snapshot copying

Snapshot copy operations within the same account and Region using the same KMS key are always incremental copies. However, if you encrypt the snapshot copy using a different KMS key, the copy is a full copy.

When you copy a snapshot across Regions or accounts, the copy is an incremental copy if the following conditions are met:

- The snapshot was copied to the destination Region or account previously.
- The most recent snapshot copy still exists in the destination Region or account.
- The most recent snapshot copy has not been archived.
- All copies of the snapshot in the destination Region or account are either unencrypted or were encrypted using the same KMS key.

> **Tip**
>
> We recommend that you tag your snapshot copies with the volume ID and creation time so that you can keep track of the most recent snapshot copy of a volume in the destination Region or account.

To see whether your snapshot copies are incremental, check the copySnapshot CloudWatch event.

Time-based copies for Amazon EBS snapshots and EBS-backed AMIs

Time-based copies can help you meet compliance or business requirements for data replication by ensuring that your EBS snapshots and EBS-backed AMIs are copied, within and across AWS Regions, in a specified timeframe. Time-based copies can also help backup administrators meet stringent disaster recovery requirements (Recovery Point Objectives and Recovery Time Objectives), and it improves development agility by ensuring predictable copying times for snapshots and EBS-backed AMIs.

With time-based snapshot and EBS-backed AMI copy operations, you specify a completion duration, between 15 minutes and 48 hours, in which the copy is to be completed. The completion duration must be specified in 15 minute increments.

Topics

- Quotas
- Determine your completion duration
- Considerations
- Monitoring
- Pricing and billing

Quotas

The following quotas apply to time-based snapshot and EBS-backed AMI copy operations:

Quota	Description	Quota value	Adjustable
Snapshot copy operation throughpu t quota	The maximum throughput that can be achieved by a single time-base d snapshot copy operation. **Note** For AMI copy operation s, the quota applies to each individua l snapshot associated with the AMI.	500 MiB/s	No
Cumulative snapshot copy throughput quota	The maximum cumulative throughput that can be achieved by concurrent time-based snapshot copy operations between a source and destinati on Region. **Note** For AMI copy operation s, each individua	2,000 MiB/s	Yes

Quota	Description	Quota value	Adjustable
	l snapshot associated with the AMI counts towards the quota.		

When you initiate a **time-based snapshot copy operation**, you specify a completion duration. The throughput used by the request is determined by the size of the snapshot data and the requested completion duration. For example, if you copy a snapshot that has 225,000 MiB (0.214 TiB) of data, and you request a completion duration of 15 minutes, the throughput is 250 MiB/s (225,000 MiB ÷ 15 minutes = 250 MiB/s).

When you initiate a **time-based AMI copy operation**, the completion duration you specify applies to each snapshot associated with the AMI. Because each snapshot can have a different size, each snapshot is copied at a different throughput to ensure that all snapshots are copied within the completion duration. For example, say you have an AMI with the following associated snapshots:

- Snapshot 1: 200,000 MiB
- Snapshot 2: 500,000 MiB
- Snapshot 3: 450,000 MiB

If you initiate a time-based copy for this AMI and specify a completion duration of 60 minutes, the request uses the following throughput:

- Snapshot 1: 55.56 MiB/s (200,000 MiB ÷ 60 minutes = 55.56 MiB/s)
- Snapshot 2: 138.89 MiB/s (500,000 MiB ÷ 60 minutes = 138.89 MiB/s)
- Snapshot 3: 125 MiB/s (450,000 MiB ÷ 60 minutes = 125 MiB/s)

This means that the request uses 319.45 MiB/s of your cumulative snapshot copy throughput quota to ensure that the copy completes in 60 minutes.

If you initiate a time-based snapshot or EBS-backed AMI copy request and your available cumulative snapshot copy throughput quota is:

- greater than or equal to the required throughput rate, the copy completes within the requested completion duration.
- less than the required throughput rate but greater than zero, the request succeeds but it will take longer than you requested. The copy is completed using your available throughput quota.
- zero (quota reached), the request fails.

Determine your completion duration

The minimum completion duration you can request for a time-based snapshot or EBS-backed AMI copy operation is 15 minutes, and the maximum completion duration you can request is 48 hours. The completion duration must be specified in 15 minute increments.

Concurrent time-based snapshot copy operations

You can perform concurrent time-based snapshot copy operations between the same source and destination Regions, as long as the combined throughput of all of the concurrent operations is within your cumulative snapshot copy throughput quota (2,000 MiB/s by default).

To determine whether you can achieve your required completion duration for your existing snapshots, divide the combined size of all of your snapshots by your required completion duration to determine the required throughput rate.

> **Tip**
>
> If you don't know the exact size of the data in your snapshots, you can use the full snapshot size as a proxy instead. To get the full snapshot size, use the describe-snapshots AWS CLI command.

```
required throughput rate = combined snapshot size ÷ required completion duration
```

If the required throughput rate is less than your cumulative snapshot copy throughput quota, you can achieve your required completion duration. If the required throughput rate is greater than your cumulative snapshot copy throughput quota, we recommend that you request a quota increase that is at least 10% higher than your required throughput rate.

> **Tip**
>
> The Amazon EC2 console provides a calculator that you can use to check how much snapshot data you copied between two Regions over a specific period, and the minimum achievable completion duration that you can achieve for that amount of data, based on a specific cumulative snapshot copy throughput quota. The calculator uses the `SnapshotCopyBytesTransferred` CloudWatch metric to calculate data copied between two Regions over a period. To open the calculator, in the Amazon EC2 console navigation panel, select **Snapshots**, and then choose **Actions**, **Launch copy duration calculator**. The snapshot copy duration calculator are not supported with AWS Outposts, Local Zones, and Wavelength Zones.

Individual time-based snapshot copy operations

You can calculate the minimum completion duration for an individual time-based snapshot copy operation by dividing the size of the snapshot data by the snapshot copy operation throughput quota (500 MiB/s).

> **Tip**
>
> If you don't know the exact size of the data in your snapshots, you can use the full snapshot size as a proxy instead. To get the full snapshot size, use the describe-snapshots AWS CLI command.

```
minimum completion duration = Max(15 minutes, (snapshot data size ÷ 500 MiB/s)
```

For example, the minimum completion duration for a snapshot with 900,000 MiB of data is 30 minutes.

```
minimum completion duration = Max(15 minutes, (900,000 MiB ÷ 500 MiB/s)
= Max(15 minutes, 30 minutes)
= 30 minutes
```

Time-based AMI copy operations

When you initiate a time-based AMI copy operation for an EBS-backed AMI with a single associated snapshot, it behaves in the same way as an **individual time-based snapshot copy operation**, and the same throughput limitations apply.

When you initiate a time-based AMI copy operation for an EBS-backed AMI with a multiple associated snapshots, it behaves in the same way as **concurrent time-based snapshot copy operations** and the same throughput limitations apply. Each associated snapshot results in a separate snapshot copy request, each of which contributes to your cumulative snapshot copy throughput quota. The completion duration that you specify applies to each associated snapshot.

Considerations

- You can initiate time-based snapshot and EBS-backed AMI copy operations when copying snapshots within the same Region or when copying snapshots across Regions.
- If you initiate two time-based copy operations for the same snapshot or AMI, the second copy operation's completion duration starts only after the first copy operation completes.
- Time-based copy operations and the snapshot copy duration calculator are not supported with AWS Outposts, Local Zones, and Wavelength Zones.

Monitoring

You can monitor the progress of time-based snapshot and EBS-backed AMI copy operations using the Amazon EC2 console and the AWS CLI. In the console, select the snapshot and then, in the **Details tab**, inspect the **Progress** field. With the AWS CLI, inspect the `Progress` output element in the describe-snapshots command response.

You can check whether a time-based snapshot or EBS-backed AMI copy operation completed within the requested completion duration by checking the difference between the **Started** and **Completed** times in the console, or `StartTime` and `CompletionTime` in the `describe-snapshots` response.

You can also use the `copySnapshot` Amazon EventBridge event to monitor the outcome of time-based copy operations. The event indicates whether the operation completed and whether the requested completion duration was met. If the completion duration was not met, the event includes more information about the cause. For more information, see EBS snapshot events.

Pricing and billing

> **Note**
>
> Similar to standard snapshot copy operations, if you copy a snapshot to a new Region, a full (non-incremental) copy is created, which results in additional storage costs. Subsequent copies of the same snapshot are incremental. Additionally, if you use external or cross-region data transfers, additional Amazon EC2 data transfer charges will apply.

Additional charges apply for time-based snapshot and EBS-backed AMI copy operations. Time-based copy operations are charged at a rate that is based on the requested completion duration, per GiB of snapshot data copied. The fixed rates are as follows:

> **Note**
>
> The completion duration must be specified in 15 minute increments. The minimum completion duration is 15 minutes, and the maximum is 48 hours.

- 15 minutes — $0.020 per GiB of data
- 30 minutes and 45 minutes — $0.018 per GiB of data
- 1 hour to 1 hour 45 minutes — $0.016 per GiB of data
- 2 hours to 3 hours 45 minutes — $0.014 per GiB of data
- 4 hours to 7 hours 45 minutes — $0.012 per GiB of data
- 8 hours to 15 hours 45 minutes — $0.010 per GiB of data
- 16 hours or more — $0.005 per GiB of data

For example, if you copy a snapshot with 3,000 GiB of data with a completion duration of 8 hours, you are billed $30 ($0.010 x 3,000 GiB).

If you initiate a time-based copy operation, but the requested completion duration is not met due to you exceeding a quota, you are billed based on the actual completion duration instead of the requested completion duration. For example, if you request a completion duration of 1 hour, but the operation completes in 2 hours, you are billed based on the rate for the 2 hour completion duration.

If Amazon EBS is not able to achieve the requested completion duration or if a request is canceled due to service-side issues, you are not billed the additional charges for the time-based snapshot copy operation.

If you delete the snapshot copy while the time-based snapshot copy operation is still in progress, you are billed for the data copied up to that point at the rate corresponding to the specified completion duration.

Encryption and snapshot copying

> **Note**
>
> Amazon S3 server-side encryption (256-bit AES) protects a snapshot's data in transit during a copy operation.

You can create an encrypted snapshot copy of a source snapshot that is unencrypted. And you can encrypt a snapshot copy with a KMS key that is different from the source snapshot. However, changing the encryption status of a snapshot copy during a copy operation could result in a full (not incremental) copy, which might incur greater data transfer and storage charges.

> **Tip**
>
> When using an encrypted snapshot that is shared with you, we recommend that you re-encrypt the snapshot by copying it and using a KMS key that you own. This protects you if the original KMS key is compromised, or if the owner revokes your access, which could cause you to lose access to the snapshot and any encrypted volumes that you created from it.

Permissions for copying encrypted snapshots

To copy an encrypted snapshot, your user must have the following permissions to use Amazon EBS encryption.

- `kms:DescribeKey`
- `kms:CreateGrant`
- `kms:GenerateDataKey`
- `kms:GenerateDataKeyWithoutPlaintext`

  - `kms:ReEncrypt`
  - `kms:Decrypt`
- To copy an encrypted snapshot that is shared from another AWS account, you must have permissions to use customer managed key that was used to encrypt that snapshot. For more information, see Share the KMS key used to encrypt a shared Amazon EBS snapshot.

Encryption outcomes for snapshot copies

The following table describes the encryption outcomes when copying snapshots that you own and snapshots that are shared with you.

Encryption by default for destination Region	Source snapshot	Snapshot copy encryption outcome	Note
Disabled	Unencrypted	Optional encryption	If you encrypt the copy, you can specify the KMS key to use. If you encrypt the copy but do not specify a KMS key, the key specified for encryption by default is used.
Disabled	Encrypted	Automatically encrypted	You can specify the KMS key to use. If you do not specify a KMS key, the AWS managed key (`aws/ebs`) is used.
Enabled	Unencrypted	Automatically encrypted	You can specify the KMS key to use. If you do not specify a KMS key, the key specified for encryption by default is used.
Enabled	Encrypted	Automatically encrypted	You can specify the KMS key to use. If you do not specify a KMS key, the key specified for encryption by default is used.

Copy a snapshot

You can copy snapshots from one Region to another. You can copy an unencrypted snapshot to an encrypted snapshot. However, if you attempt to copy an encrypted snapshot without having permissions to use the encryption key, the operation fails silently and the snapshot copy receives the "Given key ID is not accessible" status message.

Console

To copy a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot to copy, and then choose **Actions**, **Copy snapshot**.
4. For **Description**, enter a brief description for the snapshot copy.

   By default, the description includes information about the source snapshot so that you can identify a copy from the original.
5. Specify the destination for the snapshot copy.
   - To copy the snapshot to the same Region or to a different Region, select **AWS Region** and then select the destination Region.
   - To copy the snapshot to a Local Zone, select **AWS Local Zone** and then select the destination Local Zone.
   - (*Outpost customers only*) To copy the snapshot to an Outpost, select **AWS Outpost** and then enter the ARN of the destination Outpost.
6. If you need the snapshot copy to be completed within a specific timeframe, select **Enable time-based copy**. For **Completion duration**, enter the required completion duration, in 15-minute increments. For more information, Time-based copies for Amazon EBS snapshots and EBS-backed AMIs.

   If you do not need the snapshot copy to be completed in a specific timeframe, do not enable time-based copy. In this case, the snapshot copy is completed on a best-effort basis.
7. (*Outpost customers only*) To create the snapshot copy on an Outpost in the selected Region, for **Snapshot destination** choose **AWS Outpost**, and then for **Destination Outpost ARN**, enter the ARN of the Outpost to which to copy the snapshot. The **Snapshot destination** field appears only if you have and Outpost in the selected Region.
8. Specify the encryption status for the snapshot copy.

If the source snapshot is encrypted, or if your account is enabled for encryption by default, the snapshot copy is automatically encrypted. If the source snapshot is unencrypted and your account is not enabled for encryption by default, encryption is optional.

9. Choose **Copy snapshot**.

AWS CLI

To copy a snapshot to another Region

Use the copy-snapshot command. The following example copies the specified snapshot from the source Region to the current Region, which is specified by the `--region` option.

```
aws ec2 copy-snapshot \
    --source-snapshot-id snap-0abcdef1234567890 \
    --source-region us-east-1 \
    --region us-west-2
```

To copy an unencrypted snapshot to an encrypted snapshot

Use the copy-snapshot command. The following example copies the specified unencrypted snapshot from the source Region to the current Region, encrypting the new snapshot using the specified KMS key.

```
aws ec2 copy-snapshot \
    --source-snapshot-id snap-0abcdef1234567890 \
    --source-region us-east-1 \
    --encrypted \
    --kms-key-id alias/my-kms-key
```

PowerShell

To copy a snapshot to another Region

Use the Copy-EC2Snapshot cmdlet. The following example copies the specified snapshot from the source Region to the current Region, which is specified by the `--region` option.

```
 Copy-EC2Snapshot `
    -SourceSnapshotId snap-0abcdef1234567890 `
    -SourceRegion us-east-1 `
```

```
    -Region us-west-2
```

To copy an unencrypted snapshot to an encrypted snapshot

Use the Copy-EC2Snapshot cmdlet. The following example copies the specified unencrypted snapshot from the source Region to the current Region, encrypting the new snapshot using the specified KMS key.

```
Copy-EC2Snapshot `
    -SourceSnapshotId snap-0abcdef1234567890 `
    -SourceRegion us-east-1 `
    -Encrypted $true `
    -KmsKeyId alias/my-kms-key
```

Share an Amazon EBS snapshot with other AWS accounts

You can modify the permissions of a snapshot if you want to share it with other AWS accounts. You can share snapshots publicly with all other AWS accounts, or you can share them privately with individual AWS accounts that you specify. Users that you have authorized can use the snapshots that you share to create their own EBS volumes, while your original snapshot remains unaffected.

Important

When you share a snapshot, you are giving others access to all of the data on the snapshot. Share snapshots only with people that you trust with *all* of your snapshot data.

To prevent the public sharing of snapshots, you can enable Block public access for Amazon EBS snapshots.

Topics

Before you share a snapshot

The following considerations apply to sharing snapshots:

- If block public access for snapshots is enabled for the Region, attempts to publicly share snapshots will be blocked. Snapshots can still be privately shared.
- Snapshots are constrained to the Region in which they were created. To share a snapshot with another Region, copy the snapshot to that Region and then share the copy. For more information, see Copy an Amazon EBS snapshot.
- You can't share snapshots that are encrypted with the default AWS managed key. You can only share snapshots that are encrypted with a customer managed key. For more information, see Creating Keys in the *AWS Key Management Service Developer Guide*.
- You can share only unencrypted snapshots publicly.
- When you share an encrypted snapshot, you must also share the customer managed key used to encrypt the snapshot. For more information, see Share the KMS key used to encrypt a shared Amazon EBS snapshot.

Share a snapshot

You can share a snapshot publicly or with specific AWS accounts.

Console

To share a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot to share, and then choose **Actions**, **Modify permissions**.
4. Specify the snapshot's permissions. *Current setting* indicates the snapshot's current sharing permissions.
   - To share the snapshot publicly with all AWS accounts, choose **Public**.
   - To share the snapshot privately with specific AWS accounts, choose **Private**. Then, in the **Sharing accounts** section, choose **Add account**, and enter the 12-digit account ID (without hyphens) of the account to share with.
5. Choose **Save changes**.

AWS CLI

The permissions for a snapshot are specified using the `createVolumePermission` attribute of the snapshot. To make a snapshot public, set the group to `all`. To share a snapshot with a specific AWS account, set the user to the ID of the AWS account.

To share a snapshot publicly

Use the [modify-snapshot-attribute](https://docs.aws.amazon.com/cli/latest/reference/ec2/modify-snapshot-attribute.html) command.

For `--attribute`, specify `createVolumePermission`. For `--operation-type`, specify `add`. For `--group-names`, specify `all`.

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-0abcdef1234567890 \
    --attribute createVolumePermission \
    --operation-type add \
    --group-names all
```

To share a snapshot privately

Use the [modify-snapshot-attribute](https://docs.aws.amazon.com/cli/latest/reference/ec2/modify-snapshot-attribute.html) command.

For `--attribute`, specify `createVolumePermission`. For `--operation-type`, specify `add`. For `--user-ids`, specify the 12-digit IDs of the AWS accounts with which to share the snapshots.

```
aws ec2 modify-snapshot-attribute \
    --snapshot-id snap-0abcdef1234567890 \
    --attribute createVolumePermission \
    --operation-type add \
    --user-ids 123456789012 111122223333
```

PowerShell

The permissions for a snapshot are specified using the `createVolumePermission` attribute of the snapshot. To make a snapshot public, set the group to `all`. To share a snapshot with a specific AWS account, set the user to the ID of the AWS account.

To share a snapshot publicly

Use the Edit-EC2SnapshotAttribute cmdlet.

For `-Attribute`, specify `CreateVolumePermission`. For `-OperationType`, specify `Add`. For `-GroupName`, specify `all`.

```
Edit-EC2SnapshotAttribute `
    -SnapshotId snap-0abcdef1234567890 `
    -Attribute CreateVolumePermission `
    -OperationType Add `
    -GroupName all
```

To share a snapshot privately

Use the Edit-EC2SnapshotAttribute cmdlet.

For `-Attribute`, specify `CreateVolumePermission`. For `-OperationType`, specify `Add`. For `UserId`, specify the 12-digit IDs of the AWS accounts with which to share the snapshots.

```
Edit-EC2SnapshotAttribute `
    -SnapshotId snap-0abcdef1234567890 `
    -Attribute CreateVolumePermission `
    -OperationType Add `
    -UserId 123456789012 111122223333
```

Share the KMS key used to encrypt a shared Amazon EBS snapshot

When you share an encrypted snapshot, you must also share the customer managed key used to encrypt the snapshot. You can apply cross-account permissions to a customer managed key either when it is created or at a later time.

Users of your shared customer managed key who are accessing encrypted snapshots must be granted permissions to perform the following actions on the key:

- `kms:DescribeKey`
- `kms:CreateGrant`
- `kms:GenerateDataKey`
- `kms:GenerateDataKeyWithoutPlaintext`
- `kms:ReEncrypt`

- `kms:Decrypt`

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to `kms:CreateGrant`. Instead, use the `kms:GrantIsForAWSResource` condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service.

For more information about controlling access to a customer managed key, see [Using key policies in AWS KMS]() in the *AWS Key Management Service Developer Guide*.

To share customer managed key using the AWS KMS console

1. Open the AWS KMS console at https://console.aws.amazon.com/kms.
2. To change the AWS Region, use the Region selector in the upper-right corner of the page.
3. Choose **Customer managed keys** in the navigation pane.
4. In the **Alias** column, choose the alias (text link) of the customer managed key that you used to encrypt the snapshot. The key details open in a new page.
5. In the **Key policy** section, you see either the *policy view* or the *default view*. The policy view displays the key policy document. The default view displays sections for **Key administrators**, **Key deletion**, **Key Use**, and **Other AWS accounts**. The default view displays if you created the policy in the console and have not customized it. If the default view is not available, you'll need to manually edit the policy in the policy view. For more information, see [Viewing a Key Policy (Console)]() in the *AWS Key Management Service Developer Guide*.

   Use either the policy view or the default view, depending on which view you can access, to add one or more AWS account IDs to the policy, as follows:

   - (Policy view) Choose **Edit**. Add one or more AWS account IDs to the following statements: `"Allow use of the key"` and `"Allow attachment of persistent resources"`. Choose **Save changes**. In the following example, the AWS account ID 444455556666 is added to the policy.

     ```
     {
       "Sid": "Allow use of the key",
       "Effect": "Allow",
     ```

```
    "Principal": {"AWS": [
      "arn:aws:iam::111122223333:user/KeyUser",
      "arn:aws:iam::444455556666:root"
    ]},
    "Action": [
      "kms:Encrypt",
      "kms:Decrypt",
      "kms:ReEncrypt*",
      "kms:GenerateDataKey*",
      "kms:DescribeKey"
    ],
    "Resource": "*"
  },
  {
    "Sid": "Allow attachment of persistent resources",
    "Effect": "Allow",
    "Principal": {"AWS": [
      "arn:aws:iam::111122223333:user/KeyUser",
      "arn:aws:iam::444455556666:root"
    ]},
    "Action": [
      "kms:CreateGrant",
      "kms:ListGrants",
      "kms:RevokeGrant"
    ],
    "Resource": "*",
    "Condition": {"Bool": {"kms:GrantIsForAWSResource": true}}
  }
```

- (Default view) Scroll down to **Other AWS accounts**. Choose **Add other AWS accounts** and enter the AWS account ID as prompted. To add another account, choose **Add another AWS account** and enter the AWS account ID. When you have added all AWS accounts, choose **Save changes**.

Use Amazon EBS snapshots that are shared with you

To use a shared unencrypted snapshot

Locate the shared snapshot by ID or description. You can use this snapshot as you would any other snapshot that you own in your account. For example, you can create a volume from the snapshot or copy it to a different Region.

To use a shared encrypted snapshot

Locate the shared snapshot by ID or description. Create a copy of the shared snapshot in your account, and encrypt the copy with a KMS key that you own. You can then use the copy to create volumes or you can copy it to different Regions.

Console

To view snapshot permissions

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot.
4. If the filter is **Owned by me**, the snapshot is owned by this account. If the filter is **Private snapshots**, the snapshot is either owned by this account or shared specifically with this account. Select a snapshot and on the **Details** tab, check whether **Owner** specifies this account or a different account.

AWS CLI

To view snapshot permissions

Use the describe-snapshot-attribute command to get the snapshot permissions of the specified snapshot.

```
aws ec2 describe-snapshot-attribute \
    --snapshot-id snap-0abcdef1234567890 \
    --attribute createVolumePermission
```

The following is example output.

```
{
    "SnapshotId": "snap-0abcdef1234567890",
    "CreateVolumePermissions": [
        {
            "UserId": "111122223333"
        }
    ]
}
```

PowerShell

To view snapshot permissions

Use the Get-EC2SnapshotAttribute cmdlet.

```
(Get-EC2SnapshotAttribute `
    -SnapshotId snap-0abcdef1234567890 `
    -Attribute createVolumePermission).CreateVolumePermissions
```

The following is example output.

```
Group UserId
----- ------
      111122223333
```

Determine the use of snapshots that you share

You can use AWS CloudTrail to monitor whether a snapshot that you have shared with others is copied or used to create a volume. The following events are logged in CloudTrail when an action is taken on a snapshot you have shared::

- **SharedSnapshotCopyInitiated** — A shared snapshot is being copied.
- **SharedSnapshotVolumeCreated** — A shared snapshot is being used to create a volume.

For more information about using CloudTrail, see Log Amazon EC2 and Amazon EBS API calls with AWS CloudTrail.

Archive Amazon EBS snapshots

Amazon EBS Snapshots Archive is a storage tier that you can use for low-cost, long-term storage of your rarely-accessed snapshots that do not need frequent or fast retrieval.

By default, when you create a snapshot, it is stored in the Amazon EBS Snapshot Standard tier (*standard tier*). Snapshots stored in the standard tier are incremental. This means that only the blocks on the volume that have changed after your most recent snapshot are saved.

When you archive a snapshot, the incremental snapshot is converted to a full snapshot, and it is moved from the standard tier to the Amazon EBS Snapshots Archive tier (*archive tier*). Full snapshots include all of the blocks that were written to the volume at the time when the snapshot was created.

When you need to access an archived snapshot, you can restore it from the archive tier to the standard tier, and then use it in the same way that you use any other snapshot in your account.

Amazon EBS Snapshots Archive offers up to 75 percent lower snapshot storage costs for snapshots that you plan to store for 90 days or longer and that you rarely need to access.

Some typical use cases include:

- Archiving the only snapshot of a volume, such as end-of-project snapshots
- Archiving full, point-in-time incremental snapshots for compliance reasons.
- Archiving monthly, quarterly, or yearly incremental snapshots.

Topics

Quotas

This section describes the default quotas for archived and in-progress snapshots.

Quota	Default quota
Archived snapshots	25

Quota	Default quota
per volume	
Concurrent in-progress snapshot archives per account	25
Concurrent in-progress snapshot restores per account	5

If you need more than the default limits, complete the Support Center Create case form to request a limit increase.

Considerations and limitations for archiving Amazon EBS snapshots

Keep the following in mind when archiving Amazon EBS snapshots.

Considerations

- The minimum archive period is 90 days. If you delete or permanently restore an archived snapshot before the minimum archive period of 90 days, you are billed for remaining days in the archive tier, rounded to the nearest hour. For more information, see Pricing and billing for archiving Amazon EBS snapshots.
- It can take up to 72 hours to restore an archived snapshot from the archive tier to the standard tier, depending on the size of the snapshot.

- Archived snapshots are always full snapshots. A full snapshot contains all the blocks written to the volume at the time the snapshot was created. The full snapshot will likely be larger than the incremental snapshot from which it was created. However, if you have only one snapshot of a volume on the standard tier, the size of the full snapshot in the archive tier will be the same size as the snapshot in standard tier. This is because the first snapshot taken of a volume is always a full snapshot. To get the full snapshot size, use the describe-snapshots AWS CLI command.
- Archiving is recommended for monthly, quarterly, or yearly snapshots. Archiving daily incremental snapshots of a single volume can lead to higher costs when compared to keeping them in the standard tier.
- When a snapshot is archived, the data of the snapshot that is referenced by other snapshots in the snapshot lineage are retained in the standard tier. Data and storage costs associated with the referenced data that is retained on the standard tier are allocated to the next snapshot in the lineage. This ensures that subsequent snapshots in the lineage are not affected by the archival.
- If you delete an archived snapshot that matches a Recycle Bin retention rule, the archived snapshot is retained in the Recycle Bin for the retention period defined in the retention rule. To use the snapshot, you must first recover it from the Recycle Bin and then restore it from the archive tier. For more information, see Recycle Bin and Pricing and billing for archiving Amazon EBS snapshots.
- You can't use an archived snapshot in a block device mapping or to create an Amazon EBS volume.
- You can archive snapshots created by AWS Backup using the AWS Backup console, APIs, or command line tools. For more information, see Creating a backup plan in the *AWS Backup Developer Guide*.

Limitations

- You can archive snapshots that are in the `completed` state only.
- You can archive only snapshots that you own in your account. To archive a snapshot that is shared with you, first copy the snapshot to your account and then archive the snapshot copy.
- Before you can use an archived snapshot, you must first restore it to the standard tier. Restoring to the standard tier is required to create a volume from the snapshot through the `CreateVolume` and `RunInstances` API operations as well as to share or copy a snapshot. For more information, see Restore an archived Amazon EBS snapshot.
- You can archive a snapshot that is associated with one or more AMIs only if all of the associated AMIs are disabled. For more information, see Disable an AMI.

- You can't enable a disabled AMI if the associated snapshots are temporarily restored. All of the associated snapshots must be permanently restored before you can enable the AMI.
- You can't cancel the snapshot archive or snapshot restore process after it has been started.
- You can't share archived snapshots. If you archive a snapshot that you have shared with other accounts, the accounts with which the snapshot is shared lose access after the snapshot is archived.
- You can't copy an archived snapshot. If you need to copy an archived snapshot, you must first restore it.
- You can't enable fast snapshot restore for an archived snapshot. Fast snapshot restore is automatically disabled when a snapshot is archived. If you need to use fast snapshot restore, you must manually enable it after restoring the snapshot.

Pricing and billing for archiving Amazon EBS snapshots

Archived snapshots are billed at a rate of $0.0125 per GB-month. For example, if you archive a 100 GiB snapshot, you are billed $1.25 (100 GiB * $0.0125) per month.

Snapshot restores are billed at a rate of $0.03 per GB of data restored. For example, if you restore a 100 GiB snapshot from the archive tier, you are billed one time for $3 (100 GiB * $0.03).

After the snapshot is restored to the standard tier, the snapshot is billed at the standard rate for snapshots of $0.05 per GB-month.

For more information, see Amazon EBS pricing.

Billing for the minimum archive period

The minimum archive period is 90 days. If you delete or permanently restore an archived snapshot before the minimum archive period of 90 days, you are billed a pro-rated charge equal to the archive tier storage charge for the remaining days, rounded to the nearest hour. For example, if you delete or permanently restore an archived snapshot after 40 days, you are billed for the remaining 50 days of the minimum archive period.

> **Note**
>
> Temporarily restoring an archived snapshot before the minimum archive period of 90 days does not incur this charge.

Temporary restores

When you temporarily restore a snapshot, the snapshot is restored from the archive tier to the standard tier, and a copy of the snapshot remains in the archive tier. You are billed for both the snapshot in the standard tier and the snapshot copy in the archive tier for the duration of the temporary restore period. When the temporarily restored snapshot is removed from the standard tier, you are no longer billed for it, and you are billed for the snapshot in the archive tier only.

Permanent restores

When you permanently restore a snapshot, the snapshot is restored from the archive tier to the standard tier, and the snapshot is deleted from the archive tier. You are billed for the snapshot in the standard tier only.

Deleting snapshots

If you delete a snapshot while it is being archived, you are billed for the snapshot data that has already been moved to the archive tier. This data is subject to the minimum archive period of 90 days and billed accordingly upon deletion. For example, if you archive a 100 GiB snapshot, and you delete the snapshot after only 40 GiB has been archived, you are billed $1.50 for the minimum archive period of 90 days for the 40 GiB that has already been archived ($0.0125 per GB-month * 40 GB * (90 days * 24 hours) / (24 hours/day * 30-day month).

If you delete a snapshot while it is being restored from the archive tier, you are billed for the snapshot restore for the full size of the snapshot (snapshot size * $0.03). For example, if you restore a 100 GiB snapshot from the archive tier, and you delete the snapshot at any point before the snapshot restore completes, you are billed $3 (100 GiB snapshot size * $0.03).

Recycle Bin

Archived snapshots are billed at the rate for archived snapshots while they are in the Recycle Bin. Archived snapshots that are in the Recycle Bin are subject to the minimum archive period of 90 days and they are billed accordingly if they are deleted by Recycle Bin before the minimum archive period. In other words, if a retention rule deletes an archived snapshot from the Recycle Bin before the minimum period of 90 days, you are billed for the remaining days.

If you delete a snapshot that matches a retention rule while the snapshot is being archived, the archived snapshot is retained in the Recycle Bin for the retention period defined in the retention rule. It is billed at the rate for archived snapshots.

If you delete a snapshot that matches a retention rule while the snapshot is being restored, the restored snapshot is retained in the Recycle Bin for the remainder of the retention period, and billed at the standard snapshot rate. To use the restored snapshot, you must first recover it from the Recycle Bin.

For more information, see Recycle Bin.

Cost tracking

Archived snapshots appear in the AWS Cost and Usage Report with their same resource ID and Amazon Resource Name (ARN). For more information, see the AWS Cost and Usage Report User Guide.

You can use the following usage types to identify the associated costs:

- `SnapshotArchiveStorage` — fee for monthly data storage
- `SnapshotArchiveRetrieval` — one-time fee for snapshot restores
- `SnapshotArchiveEarlyDelete` — fee for deleting or permanently restoring a snapshot before the minimum archive period (90 days)

Guidelines and best practices for archiving Amazon EBS snapshots

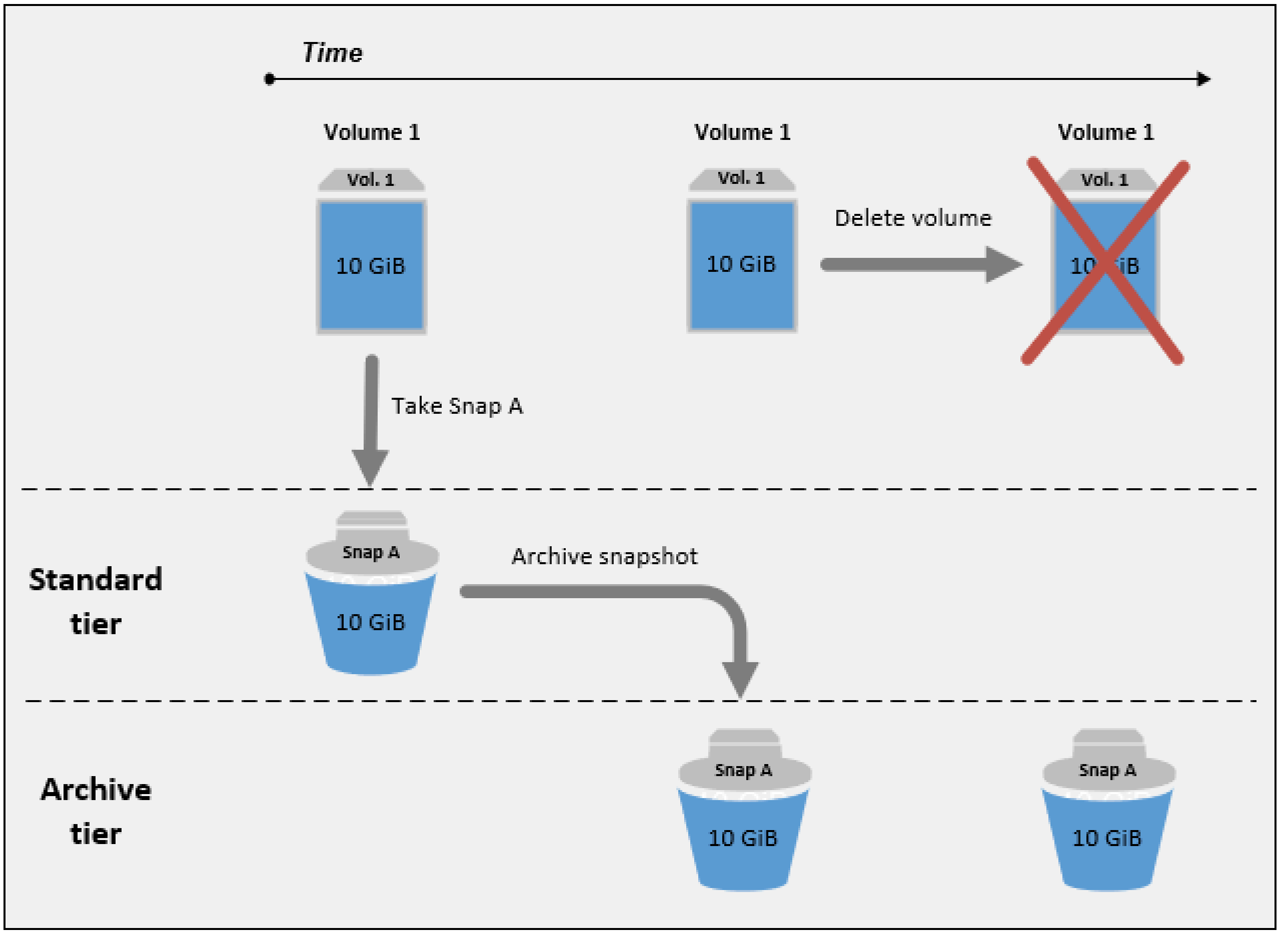
This section provides some guidelines and best practices for archiving snapshots.

Topics

Archiving the only snapshot of a volume

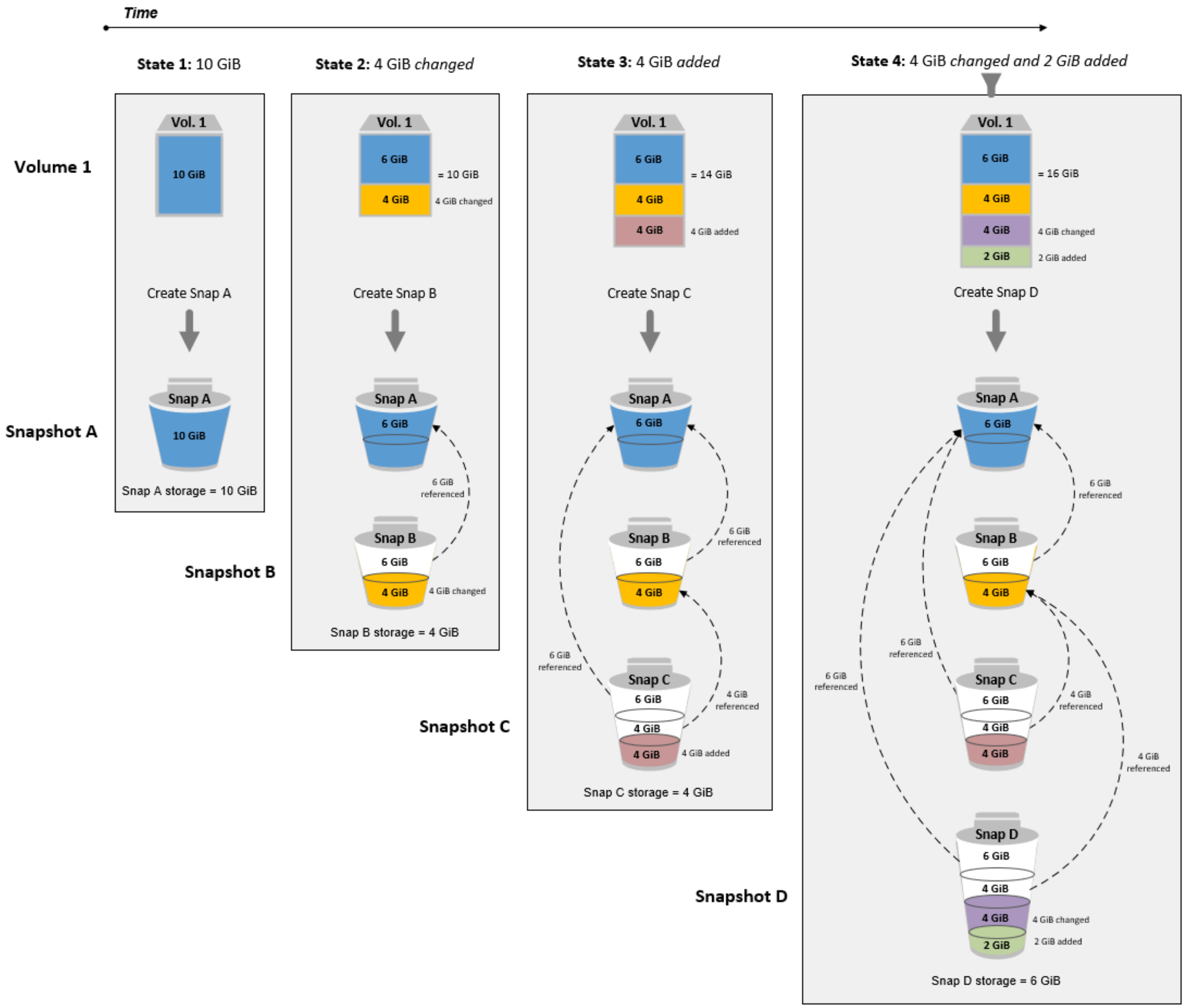
When you have only one snapshot of a volume, the snapshot is always the same size as the blocks written to the volume at the time the snapshot was created. When you archive such a snapshot, the snapshot in the standard tier is converted to an equivalent-sized full snapshot and it is moved from the standard tier to the archive tier.

Archiving these snapshots can help you save with lower storage costs. If you no longer need the source volume, you can delete the volume for further storage cost savings.



Archiving incremental snapshots of a single volume

When you archive an incremental snapshot, the snapshot is converted to a full snapshot and it is moved to the archive tier. For example, in the following image, if you archive **Snap B**, the snapshot is converted to a full snapshot that is 10 GiB in size and moved to the archive tier. Similarly, if you archive **Snap C**, the size of the full snapshot in the archive tier is 14 GiB.



If you are archiving snapshots to reduce your storage costs in the standard tier, you should not archive the first snapshot in a set of incremental snapshots. These snapshots are referenced by subsequent snapshots in the snapshot lineage. In most cases, archiving these snapshots will not reduce storage costs.

> **Note**
>
> You should not archive the last snapshot in a set of incremental snapshots. The last snapshot is the most recent snapshot taken of a volume. You will need this snapshot in the

standard tier if you want to create volumes from it in the case of a volume corruption or loss.

If you archive a snapshot that contains data that is referenced by a later snapshot in the lineage, the data storage and storage costs associated with the referenced data are allocated to the later snapshot in the lineage. In this case, archiving the snapshot will not reduce data storage or storage costs. For example, in the preceding image, if you archive **Snap B**, its 4 GiB of data is attributed to **Snap C**. In this case, your overall storage costs will increase because you incur storage costs for the full version of **Snap B** in the archive tier, and your storage costs for the standard tier remain unchanged.

If you archive **Snap C**, your standard tier storage will decrease by 4 GiB because the data is not referenced by any other snapshots later in the lineage. And your archive tier storage will increase by 14 GiB because the snapshot is converted to a full snapshot.

Archiving full snapshots for compliance reasons

You might need to create full backups of volumes on a monthly, quarterly, or yearly basis for compliance reasons. For these backups, you might need standalone snapshots without backward or forward references to other snapshots in the snapshot lineage. Snapshots archived with EBS Snapshots Archive are full snapshots, and they do not have any references to other snapshots in the lineage. Additionally, you will likely need to retain these snapshots for compliance reasons for several years. EBS Snapshots Archive makes it cost-effective to archive these full snapshots for long-term retention.

Determining the reduction in standard tier storage costs

If you want to archive an incremental snapshot to reduce your storage costs, you should consider the size of the full snapshot in the archive tier and the reduction in storage in the standard tier. This section explains how to do this.

> **Important**
>
> The API responses are data accurate at the point-in-time when the APIs are called. API responses can differ as the data associated with a snapshot changes as a result of changes in the snapshot lineage.

To determine the reduction in storage and storage costs in the standard tier, use the following steps.

1. For the snapshot that you want to archive, check the full snapshot size and the source volume from which it was created. Use the [describe-snapshots](#) command, and for `--snapshot-id`, specify the ID of the snapshot that you want to archive.

   ```
   $ aws ec2 describe-snapshots --snapshot-id snapshot_id
   ```

   The `FullSnapshotSizeInBytes` reponse value indicates the full snapshot size, in bytes, and the `VolumeId` response value indicates the ID of the source volume.

   For example, the following command returns information about snapshot `snap-09c9114207084f0d9`.

   ```
   $ aws ec2 describe-snapshots --snapshot-id snap-09c9114207084f0d9
   ```

   The following example output shows that the full snapshot size is `5678912341` bytes (5.28 GiB), and the source volume is `vol-0f3e2c292c52b85c3`.

   ```
   {
       "Snapshots": [
           {
               "Description": "",
               "Tags": [],
               "Encrypted": false,
               "VolumeId": "vol-0f3e2c292c52b85c3",
               "State": "completed",
               "VolumeSize": 8,
               "StartTime": "2021-11-16T08:29:49.840Z",
               "Progress": "100%",
               "OwnerId": "123456789012",
               "FullSnapshotSizeInBytes" : "5678912341",
               "SnapshotId": "snap-09c9114207084f0d9"
           }
       ]
   }
   ```

2. Find all of the snapshots created from the source volume. Use the describe-snapshots command. Specify the `volume-id` filter, and for the filter value, specify the volume ID from the previous step.

```
$ aws ec2 describe-snapshots --filters "Name=volume-id, Values=volume_id"
```

For example, the following command returns all snapshots created from volume `vol-0f3e2c292c52b85c3`.

```
$ aws ec2 describe-snapshots --filters "Name=volume-id,
 Values=vol-0f3e2c292c52b85c3"
```

The following is the command output, which indicates that three snapshots were created from volume `vol-0f3e2c292c52b85c3`.


```
{
    "Snapshots": [
        {
            "Description": "",
            "Tags": [],
            "Encrypted": false,
            "VolumeId": "vol-0f3e2c292c52b85c3",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-11-14T08:57:39.300Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-08ca60083f86816b0"
        },
        {
            "Description": "",
            "Tags": [],
            "Encrypted": false,
            "VolumeId": "vol-0f3e2c292c52b85c3",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-11-15T08:29:49.840Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-09c9114207084f0d9"
        },
```

```
        {
            "Description": "01",
            "Tags": [],
            "Encrypted": false,
            "VolumeId": "vol-0f3e2c292c52b85c3",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-11-16T07:50:08.042Z",
            "Progress": "100%",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-024f49fe8dd853fa8"
        }
    ]
}
```

3. Using the output from the previous command, sort the snapshots by their creation times, from oldest to newest. The `StartTime` response parameter for each snapshot indicates its creation time, in UTC time format.

   For example, the snapshots returned in the previous step arranged by creation time, from oldest to newest, is as follows:

   1. `snap-08ca60083f86816b0` (oldest – created before the snapshot that you want to archive)
   2. `snap-09c9114207084f0d9` (the snapshot to archive)
   3. `snap-024f49fe8dd853fa8` (newest – created after the snapshot that you that want to archive)

4. Identify the snapshots that were created immediately before and after the snapshot that you want to archive. In this case, you want to archive snapshot `snap-09c9114207084f0d9`, which was the second incremental snapshot created in the set of three snapshots. Snapshot `snap-08ca60083f86816b0` was created immediately before, and snapshot `snap-024f49fe8dd853fa8` was created immediately after.

5. Find the unreferenced data in the snapshot that you want to archive. First, find the blocks that are different between the snapshot that was created immediately before the snapshot that you want to archive, and the snapshot that you want to archive. Use the list-changed-blocks command. For `--first-snapshot-id`, specify the ID of the snapshot that was created immediately before the snapshot that you want to archive. For `--second-snapshot-id`, specify the ID of the snapshot that you want to archive.

```
$ aws ebs list-changed-blocks --first-snapshot-id snapshot_created_before --second-
snapshot-id snapshot_to_archive
```

For example, the following command shows the block indexes for the blocks that are different between snapshot `snap-08ca60083f86816b0` (the snapshot created before the snapshot you want to archive), and snapshot `snap-09c9114207084f0d9` (the snapshot you want to archive).

```
$ aws ebs list-changed-blocks --first-snapshot-id snap-08ca60083f86816b0 --second-
snapshot-id snap-09c9114207084f0d9
```

The following shows the command output, with some blocks omitted.


```
{
    "BlockSize": 524288,
    "ChangedBlocks": [
        {
            "FirstBlockToken": "ABgBAX6y
+WH6Rm9y5zq1VyeTCmEzGmTT0jNZG1cDirFq1rOVeFbWXsH3W4z/",
            "SecondBlockToken": "ABgBASyx0bHHBnTERu
+9USLxYK/81UT0dbHIUFqUjQUkwTwK5qkjP8NSGyNB",
            "BlockIndex": 4
        },
        {
            "FirstBlockToken": "ABgBAcfL
+EfmQmlNgstqrFnYgsAxR4SDSO4LkNLYOOChGBWcfJnpn90E9XX1",
            "SecondBlockToken": "ABgBAdX0mtX6aBAt3EBy
+8jFCESMpig7csKjbO2Ocd08m2iNJV2Ue+cRwUqF",
            "BlockIndex": 5
        },
        {
            "FirstBlockToken": "ABgBAVBaFJmbP/eRHGh7vnJlAwyiyNUi3MKZmEMxs2wC3AmM/
fc6yCOAMb65",
            "SecondBlockToken":
 "ABgBAdewWkHKTcrhZmsfM7GbaHyXD1Ctcn2nppz4wYItZRmAo1M72fpXU0Yv",
            "BlockIndex": 13
        },
        {
            "FirstBlockToken": "ABgBAQGxwuf6z095L6DpRoVRVnOqPxmx9r7Wf6O+i
+ltZ0dwPpGN39ijztLn",
```

```
            "SecondBlockToken": "ABgBAUdlitCVI7c6hGsT4ckkKCw6bMRclnV
+bKjViu/9UESTcW7CD9w4J2td",
            "BlockIndex": 14
        },
        {
            "FirstBlockToken":
 "ABgBAZBfEv4EHS1aSXTXxSE3mBZG6CNeIkwxpljzmgSHICGlFmZCyJXzE4r3",
            "SecondBlockToken":
 "ABgBAVWR7QuQQB0AP2TtmNkgS4Aec5KAQVCldnpc91zBiNmSfW9ouIlbeXWy",
            "BlockIndex": 15
        },
        .....
        {
            "SecondBlockToken": "ABgBAeHwXPL+z3DBLjDhwjdAM9+CPGV5VO5Q3rEEA
+ku50P498hjnTAgMhLG",
            "BlockIndex": 13171
        },
        {
            "SecondBlockToken":
 "ABgBAbZcPiVtLx6U3Fb4lAjRdrkJMwW5M2tiCgIp6ZZpcZ8AwXxkjVUUHADq",
            "BlockIndex": 13172
        },
        {
            "SecondBlockToken": "ABgBAVmEd/pQ9VW9hWiOujOAKcauOnUFCO
+eZ5ASVdWLXWWC04ijfoDTpTVZ",
            "BlockIndex": 13173
        },
        {
            "SecondBlockToken": "ABgBAT/jeN7w
+8ALuNdaiwXmsSfM6tOvMoLBLJ14LKvavw4IiB1d0iykWe6b",
            "BlockIndex": 13174
        },
        {
            "SecondBlockToken": "ABgBAXtGvUhTjjUqkwKXfXzyR2GpQei/
+pJSG/19ESwvt7Hd8GHaUqVs6Zf3",
            "BlockIndex": 13175
        }
    ],
    "ExpiryTime": 1637648751.813,
    "VolumeSize": 8
}
```

Next, use the same command to find blocks that are different between the snapshot that you want to archive and the snapshot that was created immediately after it. For `--first-snapshot-id`, specify the ID of the snapshot that you want to archive. For `--second-snapshot-id`, specify the ID of the snapshot that was created immediately after the snapshot that you want to archive.

```
$ aws ebs list-changed-blocks --first-snapshot-id snapshot_to_archive --second-snapshot-id snapshot_created_after
```

For example, the following command shows the block indexes of the blocks that are different between snapshot `snap-09c9114207084f0d9` (the snapshot that you want to archive) and snapshot `snap-024f49fe8dd853fa8` (the snapshot created after the snapshot that you want to archive).

```
$ aws ebs list-changed-blocks --first-snapshot-id snap-09c9114207084f0d9 --second-snapshot-id snap-024f49fe8dd853fa8
```

The following shows the command output, with some blocks omitted.


```
{
    "BlockSize": 524288,
    "ChangedBlocks": [
        {
            "FirstBlockToken": "ABgBAVax0bHHBnTERu
+9USLxYK/81UT0dbSnkDk0gqwRFSFGWA7HYbkkAy5Y",
            "SecondBlockToken":
 "ABgBASEvi9x8Om7Htp37cKG2NT9XUzEbLHpGcayelomSoHpGy8LGyvG0yYfK",
            "BlockIndex": 4
        },
        {
            "FirstBlockToken": "ABgBAeL0mtX6aBAt3EBy+8jFCESMpig7csfMrI4ufnQJT3XBm/
pwJZ1n2Uec",
            "SecondBlockToken": "ABgBAXmUTg6rAI
+v0LvekshbxCVpJjWILvxgC0AG0GQBEUNRVHkNABBwXLkO",
            "BlockIndex": 5
        },
        {
            "FirstBlockToken":
 "ABgBATKwWkHKTcrhZmsfM7GbaHyXD1CtcnjIZv9YzisYsQTMHfTfh4AhS0s2",
```

```
            "SecondBlockToken": "ABgBAcmiPFovWgXQio
+VBrxOqGy4PKZ9SAAHaZ2HQBM9fQQU0+EXxQjVGv37",
            "BlockIndex": 13
        },
        {
            "FirstBlockToken":
 "ABgBAbRlitCVI7c6hGsT4ckkKCw6bMRclnARrMt1hUbIhFnfz8kmUaZOP2ZE",
            "SecondBlockToken": "ABgBAXe935n544+rxhJ0INB8q7pAeoPZkkD27vkspE/
qKyvOwpozYII6UNCT",
            "BlockIndex": 14
        },
        {
            "FirstBlockToken": "ABgBAd+yxCO26I
+1Nm2KmuKfrhjCkuaP6LXuol3opCNk6+XRGcct4suBHje1",
            "SecondBlockToken": "ABgBAcPpnXz821NtTvWBPTz8uUFXnS8jXubvghEjZulIjHgc
+7saWys77shb",
            "BlockIndex": 18
        },
        .....
        {
            "SecondBlockToken": "ABgBATni4sDE5rS8/a9pqV03lU/lKCW
+CTxFl3cQ5p2f2h1njpuUiGbqKGUa",
            "BlockIndex": 13190
        },
        {
            "SecondBlockToken": "ABgBARbXo7zFhu7IEQ/9VMYFCTCtCuQ
+iSlWVpBIshmeyeS5FD/M0i64U+a9",
            "BlockIndex": 13191
        },
        {
            "SecondBlockToken": "ABgBAZ8DhMk+rROXa4dZlNK45rMYnVIGGSyTeiMli/sp/
JXUVZKJ9sMKIsGF",
            "BlockIndex": 13192
        },
        {
            "SecondBlockToken":
 "ABgBATh6MBVE904l6sqOC27s1nVntFUpDwiMcRWGyJHy8sIgGL5yuYXHAVty",
            "BlockIndex": 13193
        },
        {
            "SecondBlockToken":
 "ABgBARuZykaFBWpCWrJPXaPCneQMbyVgnITJqj4c1kJWPIj5Gn61OQyy+giN",
            "BlockIndex": 13194
        }
```

```
    ],
    "ExpiryTime": 1637692677.286,
    "VolumeSize": 8
}
```

6. Compare the output returned by both commands in the previous step. If the same block index appears in both command outputs, it indicates that the block contains unreferenced data.

   For example, the command output in the previous step indicates that blocks 4, 5, 13, and 14 are unique to snapshot `snap-09c9114207084f0d9` and that they are not referenced by any other snapshots in the snapshot lineage.

   To determine the reduction in standard tier storage, multiply the number of blocks that appear in both command outputs by 512 KiB, which is the snapshot block size.

   For example, if 9,950 block indexes appear in both command outputs, it indicates that you will decrease standard tier storage by around 4.85 GiB (9,950 blocks * 512 KiB = 4.85 GiB).

7. Determine the storage costs for storing the unreferenced blocks in the standard tier for 90 days. Compare this value with the cost of storing the full snapshot, described in step 1, in the archive tier. You can determine your costs savings by comparing the values, assuming that you do not restore the full snapshot from the archive tier during the minimum 90-day period. For more information, see Pricing and billing for archiving Amazon EBS snapshots.

Required IAM permissions for archiving Amazon EBS snapshots

By default, users don't have permission to use snapshot archiving. To allow users to use snapshot archiving, you must create IAM policies that grant permission to use specific resources and API actions. For more information, see Creating IAM policies in the IAM User Guide.

To use snapshot archiving, users need the following permissions.

- `ec2:DescribeSnapshotTierStatus`
- `ec2:ModifySnapshotTier`
- `ec2:RestoreSnapshotTier`

Console users might need additional permissions such as `ec2:DescribeSnapshots`.

To archive and restore encrypted snapshots, the following additional AWS KMS permissions are required.

- kms:CreateGrant
- kms:Decrypt
- kms:DescribeKey

The following is an example IAM policy that gives IAM users permission to archive, restore, and view encrypted and unencrypted snapshots. It includes the ec2:DescribeSnapshots permission for console users. If some permissions are not needed, you can remove them from the policy.

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to kms:CreateGrant. Instead, use the kms:GrantIsForAWSResource condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "ec2:DescribeSnapshotTierStatus",
            "ec2:ModifySnapshotTier",
            "ec2:RestoreSnapshotTier",
            "ec2:DescribeSnapshots",
            "kms:CreateGrant",
            "kms:Decrypt",
            "kms:DescribeKey"
        ],
        "Resource": "*",
        "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                }
            }
    }]
}
```

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Archive an Amazon EBS snapshot

You can archive any snapshot that is in the `completed` state and that you own in your account. You can't archive snapshots that are in the `pending` or `error` states, or snapshots that are shared with you. For more information, see Considerations and limitations for archiving Amazon EBS snapshots.

If the snapshot is associated with one or more AMIs, then you must first disable those associated AMIs before you can archive the snapshot. For more information, see Disable an AMI.

Archived snapshots retain their snapshot ID, encryption status, AWS Identity and Access Management (IAM) permissions, owner information, and resource tags. However, fast snapshot restore and snapshot sharing are automatically disabled after the snapshot is archived.

You can continue to use the snapshot while the archive is in process. As soon as the snapshot tiering status reaches the `archival-complete` state, you can no longer use the snapshot.

Console

To archive a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.

3. In the list of snapshots, select the snapshot to archive and then choose **Actions**, **Archive snapshot**.
4. To confirm, choose **Archive snapshot**.

AWS CLI

To archive a snapshot

Use the modify-snapshot-tier AWS CLI command. For `--snapshot-id`, specify the ID of the snapshot to archive. For `--storage-tier`, specify `archive`.

```
aws ec2 modify-snapshot-tier \
    --snapshot-id snap-0abcdef1234567890 \
    --storage-tier archive
```

The following is example output. The `TieringStartTime` response parameter indicates the date and time at which the archive process was started, in UTC time format (YYYY-MM-DDTHH:MM:SSZ).

```
{
    "SnapshotId": "snap-0abcdef1234567890",
    "TieringStartTime": "2021-09-15T16:44:37.574Z"
}
```

PowerShell

To archive a snapshot

Use the Edit-EC2SnapshotTier cmdlet. The following example archives the specified snapshot.

```
Edit-EC2SnapshotTier `
    -SnapshotId snap-0abcdef1234567890 `
    -StorageTier "archive"
```

Restore an archived Amazon EBS snapshot

Before you can use an archived snapshot, you must first restore it to the standard tier. The restored snapshot has the same snapshot ID, encryption status, IAM permissions, owner information, and resource tags that it had before it was archived. After it is restored, you can use it in the same way that you use any other snapshot in your account. The restored snapshot is always a full snapshot.

When you restore a snapshot, you can choose to restore it **permanently** or **temporarily**.

If you restore a snapshot permanently, the snapshot is moved from the archive tier to the standard tier permanently. The snapshot remains restored and ready for use until you manually re-archive it or you manually delete it. When you permanently restore a snapshot, the snapshot is removed from the archive tier.

If you restore a snapshot temporarily, the snapshot is copied from the archive tier to the standard tier for a restore period that you specify. The snapshot remains restored and ready for use for the restore period only. During the restore period, a copy of the snapshot remains in the archive tier. After the period expires, the snapshot is automatically removed from the standard tier. You can increase or decrease the restore period or change the restore type to permanent at any time during the restore period. For more information, see Modify the restore period for a temporarily restored Amazon EBS snapshot.

If you are restoring snapshots that are associated with a disabled AMI, and you intend to use that AMI, you must first **permanently restore** all of the associated snapshots and then re-enable a disabled AMI before you can use it. You can't enable an AMI if the associated snapshots are temporarily restored. You can use the following command to find all of the snapshots associated with an AMI.

```
aws ec2 describe-images --image-id ami_id \
  --query Images[*].BlockDeviceMappings[*].Ebs[].SnapshotId[]
```

Console

To restore a snapshot from the archive

Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.

1. In the navigation pane, choose **Snapshots**.
2. In the list of snapshots, select the archived snapshot to restore, and then choose **Actions**, **Restore snapshot from archive**.
3. Specify the type of restore to perform. For **Restore type**, do one of the following:

   - To restore the snapshot permanently, select **Permanent**.

- To restore the snapshot temporarily, select **Temporary**, and then for **Temporary restore period**, enter the number of days for which to restore the snapshot.

4. To confirm, choose **Restore snapshot**.

AWS CLI

To permanently restore an archived snapshot

Use the restore-snapshot-tier AWS CLI command with the `--permanent-restore` option. For `--snapshot-id`, specify the ID of the snapshot to restore.

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-0abcdef1234567890 \
    --permanent-restore
```

The following is example output.

```
{
    "SnapshotId": "snap-0abcdef1234567890",
    "IsPermanentRestore": true
}
```

To temporarily restore an archived snapshot

Use the restore-snapshot-tier AWS CLI command. Omit the `--permanent-restore` option. For `--temporary-restore-days`, specify the number of days for which to restore the snapshot. The allowed range is 1 to 180 days. If you do not specify a value, the default is 1 day.

The following example temporarily restores the specified snapshot for a restore period of 5 days.

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-0abcdef1234567890 \
    --temporary-restore-days 5
```

The following is example output.

```
{
    "SnapshotId": "snap-0abcdef1234567890",
```

```
    "RestoreDuration": 5,
    "IsPermanentRestore": false
}
```

PowerShell

To permanently restore an archived snapshot

Use the Restore-EC2SnapshotTier cmdlet.

```
Restore-EC2SnapshotTier `
    -SnapshotId snap-0abcdef1234567890 `
    -PermanentRestore $true
```

To temporarily restore an archived snapshot

Use the Restore-EC2SnapshotTier cmdlet.

```
Restore-EC2SnapshotTier `
    -SnapshotId snap-0abcdef1234567890 `
    -TemporaryRestoreDays 5
```

Modify the restore period for a temporarily restored Amazon EBS snapshot

When you restore a snapshot temporarily, you must specify the number of days for which the snapshot is to remain restored in your account. After the restore period expires, the snapshot is automatically removed from the standard tier.

You can change the restore period for a temporarily restored snapshot at any time.

You can choose to either increase or decrease the restore period, or you can change the restore type from temporary to permanent.

If you change the restore period, the new restore period is effective from the current date. For example, if you specify a new restore period of 5 days, the snapshot will remain restored for five days from the current date.

> **Note**
>
> You can end a temporary restore early by setting the restore period to 1 day.

If you change the restore type from temporary to permanent, the snapshot copy is deleted from the archive tier, and the snapshot remains available in your account until you manually re-archive it or delete it.

Console

To modify the restore period or restore type

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. In the list of snapshots, select the snapshot that you previously temporarily restored, and then choose **Actions**, **Restore snapshot from archive**.
4. For **Restore type**, do one of the following:
   - To change the restore type from temporary to permanent, select **Permanent**.
   - To increase or decrease the restore period, keep **Temporary**, and then for **Temporary restore period**, enter the new restore period in days.
5. To confirm, choose **Restore snapshot**.

AWS CLI

To modify the restore period or change the restore type

Use the restore-snapshot-tier command. For `--snapshot-id`, specify the ID of the snapshot that you previously temporarily restored. To change the restore type from temporary to permanent, specify `--permanent-restore` and omit `--temporary-restore-days`. To increase or decrease the restore period, omit `--permanent-restore` and for `--temporary-restore-days`, specify the new restore period in days.

Example: Increase or decrease the restore period

The following command changes the restore period for the specified snapshot to `10` days.

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-0abcdef1234567890 \
    --temporary-restore-days 10
```

The following is example output.

```
{
    "SnapshotId": "snap-0abcdef1234567890",
    "RestoreDuration": 10,
    "IsPermanentRestore": false
}
```

Example: Change restore type to permanent

The following command changes the restore type for the specified snapshot from temporary to permanent.

```
aws ec2 restore-snapshot-tier \
    --snapshot-id snap-0abcdef1234567890 \
    --permanent-restore
```

The following is example output.

```
{
    "SnapshotId": "snap-0abcdef1234567890",
    "IsPermanentRestore": true
}
```

PowerShell

To modify the restore period or change the restore type

Use the Restore-EC2SnapshotTier cmdlet. For `-SnapshotId`, specify the ID of the snapshot that you previously temporarily restored. To change the restore type from temporary to permanent, specify `-PermanentRestore` and omit `-TemporaryRestoreDays`. To increase or decrease the restore period, omit `-PermanentRestore` and for `-TemporaryRestoreDays`, specify the new restore period in days.

Example: Increase or decrease the restore period

The following command changes the restore period for the specified snapshot to `10` days.

```
Restore-EC2SnapshotTier `
    -SnapshotId snap-0abcdef1234567890 `
    -TemporaryRestoreDays 10
```

Example: Change restore type to permanent

The following command changes the restore type for the specified snapshot from temporary to permanent.

```
Restore-EC2SnapshotTier `
    -SnapshotId snap-0abcdef1234567890 `
    -PermanentRestore $true
```

View archived Amazon EBS snapshots

Console

To view storage tier information for a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. In the list of snapshots, select the snapshot and choose the **Storage tier** tab.

   The tab provides the following information:

   - **Last tier change started on** — The date and time when the last archive or restore was started.
   - **Tier change progress** — The progress of the last archive or restore action, as a percentage.
   - **Storage tier** — The storage tier for the snapshot. Always `archive` for archived snapshots, and `standard` for snapshots stored on the standard tier, including temporarily restored snapshots.
   - **Tiering status** — The status of the last archive or restore action.
   - **Archive completed on** — The date and time when the archive completed.
   - **Temporary restore expires on** — The date and time when a temporarily restored snapshot is set to expire.

AWS CLI

To view archival information about an archived snapshot

Use the describe-snapshot-tier-status command. Specify the `snapshot-id` filter, and for the filter value, specify the snapshot ID. Alternatively, to view all archived snapshots, omit the filter.

```
--filters "Name=snapshot-id, Values=snapshot_id"
```

The output includes the following response parameters:

- `Status` — The status of the snapshot. Always `completed` for archived snapshots. Only snapshots that are in the `completed` state can be archived.
- `LastTieringStartTime` — The date and time that the archival process started, in UTC time format (YYYY-MM-DDTHH:MM:SSZ).
- `LastTieringOperationState` — The current state of the archival process. Possible states include: `archival-in-progress` | `archival-completed` | `archival-failed` | `permanent-restore-in-progress` | `permanent-restore-completed` | `permanent-restore-failed` | `temporary-restore-in-progress` | `temporary-restore-completed` | `temporary-restore-failed`
- `LastTieringProgress` — The progress of the snapshot archival process, as a percentage.
- `StorageTier` — The storage tier for the snapshot. Always `archive` for archived snapshots, and `standard` for snapshots stored on the standard tier, including temporarily restored snapshots.
- `ArchivalCompleteTime` — The date and time that the archival process completed, in UTC time format (YYYY-MM-DDTHH:MM:SSZ).

Example: Describe a snapshot

The following example displays information about the specified snapshot.

```
aws ec2 describe-snapshot-tier-status \
    --filters "Name=snapshot-id, Values=snap-0abcdef1234567890"
```

The following is example output.

```
{
    "SnapshotTierStatuses": [
        {
            "Status": "completed",
            "ArchivalCompleteTime": "2021-09-15T17:33:16.147Z",
            "LastTieringProgress": 100,
            "Tags": [],
            "VolumeId": "vol-01234567890abcdef",
```

```
            "LastTieringOperationState": "archival-completed",
            "StorageTier": "archive",
            "OwnerId": "123456789012",
            "SnapshotId": "snap-0abcdef1234567890",
            "LastTieringStartTime": "2021-09-15T16:44:37.574Z"
        }
    ]
}
```

To view archived and standard tier snapshots

Use the describe-snapshots command. For `--snapshot-ids`, specify the ID of the snapshot.

```
aws ec2 describe-snapshots --snapshot-ids snap-0abcdef1234567890
```

The following is example output. The `StorageTier` response parameter indicates whether the snapshot is currently archived. `archive` indicates that the snapshot is currently archived and stored in the archive tier, and `standard` indicates that the snapshot is currently not archived and that it is stored in the standard tier.

In the following example output, only `Snap A` is archived. `Snap B` and `Snap C` are not archived.

Additionally, the `RestoreExpiryTime` response parameter is returned only for snapshots that are temporarily restored from the archive. It indicates when temporarily restored snapshots are to be automatically removed from the standard tier. It is **not** returned for snapshots that are permanently restored.

In the following example output, `Snap C` is temporarily restored, and it will be automatically removed from the standard tier at 2021-09-19T21:00:00.000Z (September 19, 2021 at 21:00 UTC).

```
{
    "Snapshots": [
        {
            "Description": "Snap A",
            "Encrypted": false,
            "VolumeId": "vol-01234567890aaaaaa",
            "State": "completed",
            "VolumeSize": 8,
            "StartTime": "2021-09-07T21:00:00.000Z",
```

```
                "Progress": "100%",
                "OwnerId": "123456789012",
                "SnapshotId": "snap-01234567890aaaaaa",
                "StorageTier": "archive",
                "Tags": []
            },
            {
                "Description": "Snap B",
                "Encrypted": false,
                "VolumeId": "vol-09876543210bbbbbb",
                "State": "completed",
                "VolumeSize": 10,
                "StartTime": "2021-09-14T21:00:00.000Z",
                "Progress": "100%",
                "OwnerId": "123456789012",
                "SnapshotId": "snap-09876543210bbbbbb",
                "StorageTier": "standard",
                "RestoreExpiryTime": "2019-09-19T21:00:00.000Z",
                "Tags": []
            },
            {
                "Description": "Snap C",
                "Encrypted": false,
                "VolumeId": "vol-054321543210cccccc",
                "State": "completed",
                "VolumeSize": 12,
                "StartTime": "2021-08-01T21:00:00.000Z",
                "Progress": "100%",
                "OwnerId": "123456789012",
                "SnapshotId": "snap-054321543210cccccc",
                "StorageTier": "standard",
                "Tags": []
            }
        ]
}
```

To view only snapshots that are stored in the archive tier or the standard tier

Use the describe-snapshots command. Include the `--filter` option, with the filter name `storage-tier`. For the filter value specify either `archive` or `standard`. The following example displays only archived snapshots.

```
aws ec2 describe-snapshots --filters "Name=storage-tier,Values=archive"
```

PowerShell

To view archival information about an archived snapshot

Use the Get-EC2SnapshotTierStatus cmdlet.

```
Get-EC2SnapshotTierStatus `
    -Filter @{Name="snapshot-id"; Values="snap-0abcdef1234567890"}
```

To view archived and standard tier snapshots

Use the Get-EC2Snapshot cmdlet.

```
Get-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890
```

To view only snapshots that are stored in the archive tier or the standard tier

Use the Get-EC2Snapshot cmdlet. The following example displays only archived snapshots.

```
Get-EC2Snapshot `
    -Filter @{Name="storage-tier"; Values="archive"}
```

Monitor Amazon EBS snapshot archiving using CloudWatch Events

Amazon EBS emits events related to snapshot archiving actions. You can use AWS Lambda and Amazon CloudWatch Events to handle event notifications programmatically. Events are emitted on a best effort basis. For more information, see the Amazon EventBridge User Guide.

The following events are available:

- `archiveSnapshot` — Emitted when a snapshot archive action succeeds or fails.

  The following is an example of an event that is emitted when a snapshot archive action succeeds.

```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
   "detail-type": "EBS Snapshot Notification",
   "source": "aws.ec2",
```

```
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "archiveSnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "123456789",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
```

The following is an example of an event that is emitted when a snapshot archive action fails.

```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "archiveSnapshot",
    "result": "failed",
    "cause": "Source snapshot ID is not valid",
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
}
```

- `permanentRestoreSnapshot` — Emitted when a permanent restore action succeeds or fails.

The following is an example of an event that is emitted when a permanent restore action succeeds.

```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "permanentRestoreSnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-10-45T15:30:00Z"
  }
}
```

The following is an example of an event that is emitted when a permanent restore action fails.

```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "permanentRestoreSnapshot",
    "result": "failed",
    "cause": "Source snapshot ID is not valid",
```

```
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
}
```

- `temporaryRestoreSnapshot` — Emitted when a temporary restore action succeeds or fails.

  The following is an example of an event that is emitted when a temporary restore action succeeds.

```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "temporaryRestoreSnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
    "restoreExpiryTime": "2021-06-45T15:30:00Z",
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
}
```

  The following is an example of an event that is emitted when a temporary restore action fails.

```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
```

```
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "temporaryRestoreSnapshot",
    "result": "failed",
    "cause": "Source snapshot ID is not valid",
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
}
```


- `restoreExpiry` — Emitted when the restore period for a temporarily restored snapshot expires.

  The following is an example.


```
{
   "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2021-05-25T13:12:22Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "restoryExpiry",
    "result": "succeeded",
    "cause": "",
    "request-id": "1234567890",
    "snapshot_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
    "startTime": "2021-05-25T13:12:22Z",
    "endTime": "2021-05-45T15:30:00Z",
```

```
    "recycleBinExitTime": "2021-10-45T15:30:00Z"
  }
}
```

Delete an Amazon EBS snapshot

After you no longer need an Amazon EBS snapshot of a volume, you can delete it. Deleting a snapshot has no effect on the volume. Deleting a volume has no effect on the snapshots made from it.

Topics

Considerations for deleting snapshots

The following considerations apply to deleting snapshots:

- You can't delete a snapshot of the root device of an EBS volume used by a registered AMI. This consideration applies even if the registered AMI is deprecated or disabled. You must first deregister the AMI before you can delete the snapshot. For more information, see Deregister your AMI.
- You can't delete a snapshot that is managed by the AWS Backup service using Amazon EC2. Instead, use AWS Backup to delete the corresponding recovery points in the backup vault. For more information, see Deleting backups in the *AWS Backup Developer Guide*.
- You can create, retain, and delete snapshots manually, or you can use Amazon Data Lifecycle Manager to manage your snapshots for you. For more information, see Amazon Data Lifecycle Manager.
- Although you can delete a snapshot that is still in progress, the snapshot must complete before the deletion takes effect. This might take a long time. If you are also at your concurrent snapshot limit, and you attempt to take an additional snapshot, you might get a `ConcurrentSnapshotLimitExceeded` error. For more information, see the Service Quotas for Amazon EBS in the *Amazon Web Services General Reference*.

- If you delete a snapshot that matches a Recycle Bin retention rule, the snapshot is retained in the Recycle Bin instead of being immediately deleted. For more information, see Recycle Bin.
- You can't delete snapshots associated with disabled EBS-backed AMIs. For more information, see Disable an AMI.
- You can't delete snapshots that are shared with you.
- If you delete a shared snapshot that you own, all accounts with which the snapshot is shared lose access to it.

How deleting incremental snapshots works

If you make periodic snapshots of a volume, the snapshots are *incremental*. This means that only the blocks on the device that have changed after your most recent snapshot are saved in the new snapshot. Even though snapshots are saved incrementally, the snapshot deletion process is designed so that you need to retain only the most recent snapshot in order to create volumes.

If data was present on a volume held in an earlier snapshot or series of snapshots, and that data is subsequently deleted from the volume later on, the data is still considered to be unique data of the earlier snapshots. Unique data is only deleted from the sequence of snapshots if all snapshots that reference the unique data are deleted.

When you delete a snapshot, only the data that is referenced exclusively by that snapshot is removed. Unique data is only deleted if all of the snapshots that reference it are deleted. Deleting previous snapshots of a volume does not affect your ability to create volumes from later snapshots of that volume.

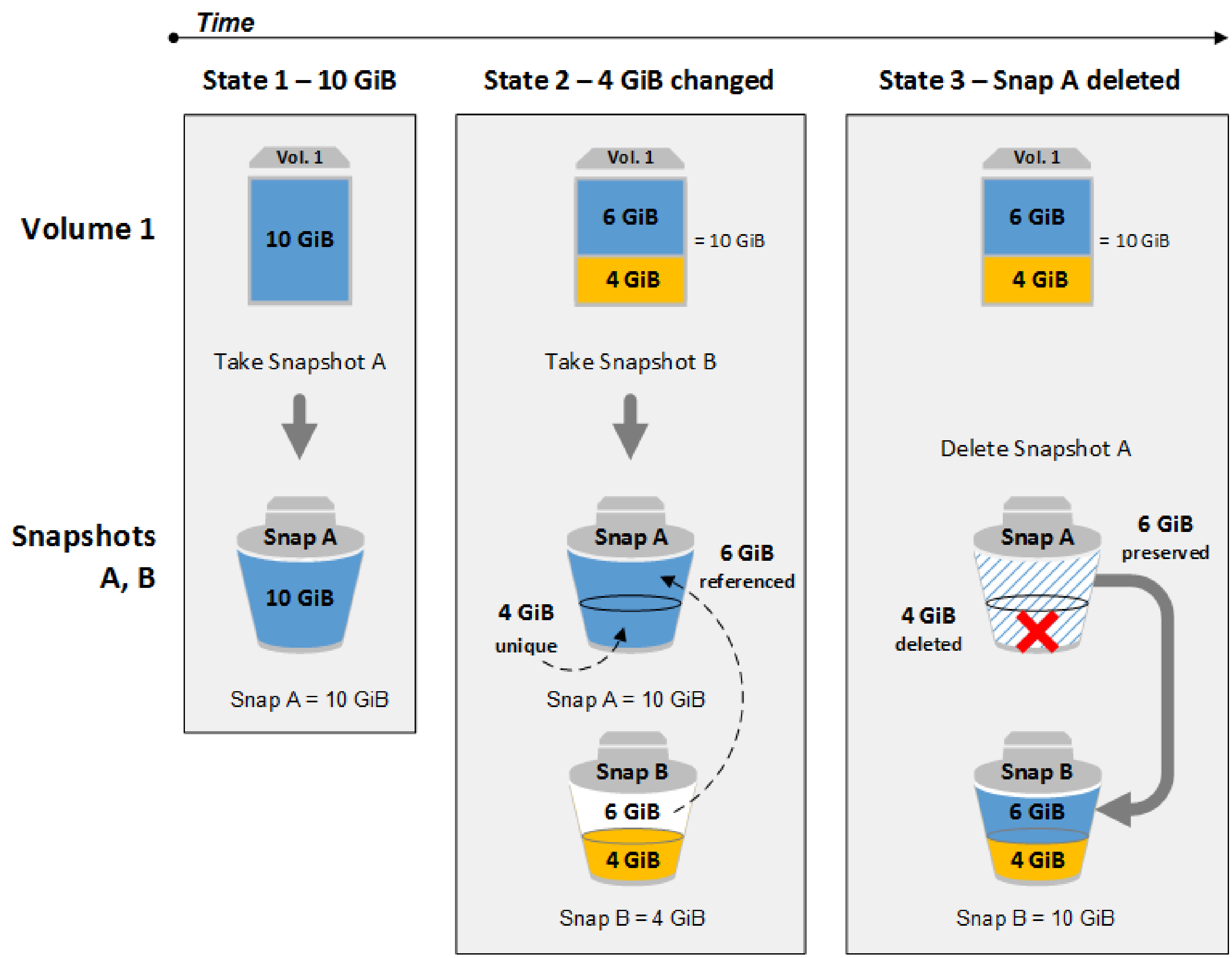
Deleting a snapshot might not reduce your organization's data storage costs. Other snapshots might reference that snapshot's data, and referenced data is always preserved. If you delete a snapshot containing data being used by a later snapshot, costs associated with the referenced data are allocated to the later snapshot. For more information about how snapshots store data, see How Amazon EBS snapshots work and the following example.

In the following diagram, Volume 1 is shown at three points in time. A snapshot has captured each of the first two states, and in the third, a snapshot has been deleted.

- In **state 1**, the volume has 10 GiB of data. Because Snap A is the first snapshot taken of the volume, the entire 10 GiB of data must be copied. In this state, you are charged for storing 10 GiB of snapshot data.

- In **state 2**, the volume still contains 10 GiB of data, but 4 GiB have changed. Snap B stores only the 4 GiB that changed after Snap A was taken, and it references the 6 GiB of unchanged data that is already stored in Snap A. In this state, you are charged for storing 14 GiB of snapshot data (10 Gib from Snap A + 4 GiB from Snap B).
- In **state 3**, the volume is unchanged but Snap A is deleted. Since the 6 GiB of unchanged data in Snap A is still referenced by Snap B, that data is retained and it is associated with Snap B. The 4 GiB of unique data in Snap A is deleted since it is no longer referenced by other snapshots. In this state, you are charged for storing 10 GiB of snapshot data (6 GiB of data retained from Snap A + 4 GiB of data in Snap B).

Deleting a snapshot with some of its data referenced by another snapshot



Delete a snapshot

Console

To delete a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot.
4. Choose **Actions**, **Delete snapshot**.
5. When prompted for confirmation, enter **`delete`** and then choose **Delete**.

AWS CLI

To delete a snapshot

Use the delete-snapshot command.

```
aws ec2 delete-snapshot --snapshot-id snap-0abcdef1234567890
```

PowerShell

To delete a snapshot

Use the Remove-EC2Snapshot cmdlet.

```
Remove-EC2Snapshot -SnapshotId snap-0abcdef1234567890
```

Troubleshooting tip

If you get a `Failed to delete snapshot` error indicating that the snapshot is currently in use by an AMI, you must deregister the associated AMI before you can delete the snapshot. You can't delete snapshots that are associated with an AMI.
If you're using the console and the associated AMI is disabled, you must select the **Disabled images** filter on the **AMIs** screen to view disabled AMIs.

Delete multi-volume snapshots

To delete multi-volume snapshots, retrieve all of the snapshots for your multi-volume snapshot set using the tag you applied to the set when you created the snapshots. Then, delete the snapshots individually.

You will not be prevented from deleting individual snapshots in the multi-volume snapshot set. If you delete a snapshot while it is in the `pending state`, only that snapshot is deleted. The other snapshots in the multi-volume snapshot set still complete successfully.

Amazon EBS fast snapshot restore

Amazon EBS fast snapshot restore (FSR) enables you to create a volume from a snapshot that is fully initialized at creation. This eliminates the latency of I/O operations on a block when it is accessed for the first time. Volumes that are created using fast snapshot restore instantly deliver all of their provisioned performance.

To get started, enable fast snapshot restore for specific snapshots in specific Availability Zones. Each snapshot and Availability Zone pair refers to one fast snapshot restore. When you create a volume from one of these snapshots in one of its enabled Availability Zones, the volume is restored using fast snapshot restore.

You must explicitly enable fast snapshot restore for each snapshot. For example, if you create a new snapshot from a volume that was restored from a fast snapshot restore-enabled snapshot, the new snapshot is not automatically enabled for fast snapshot restore. If you copy a snapshot that is enabled for fast snapshot restore, the snapshot copy is not automatically enabled for fast snapshot restore.

The number of volumes that you can restore with the full performance benefit of fast snapshot restore is determined by volume creation credits for the snapshot. For more information see Amazon EBS fast snapshot restore volume creation credits.

You can enable fast snapshot restore for snapshots that you own and for public and private snapshots that are shared with you.

Contents

- Amazon EBS fast snapshot restore volume creation credits
- Configure fast snapshot restore for an Amazon EBS snapshot
- Check the fast snapshot restore state for an Amazon EBS snapshot
- View Amazon EBS volumes restored using fast snapshot restore

Considerations

- Fast snapshot restore is not supported with AWS Outposts, Local Zones, and Wavelength Zones.
- Fast snapshot restore can be enabled on snapshots with a size of 16 TiB or less.
- Volumes provisioned with performance up to 64,000 IOPS and 1,000 MiB/s throughput receive the full performance benefit of fast snapshot restore. For volumes provisioned with performance greater than 64,000 IOPS or 1,000 MiB/s throughput, we recommend that you initialize the volume to receive its full performance.
- You can enable up to 5 snapshots for fast snapshot restore per Region. The quota applies to snapshots that you own and snapshots that are shared with you. If you enable fast snapshot restore for a snapshot that is shared with you, it counts towards your fast snapshot restore quota. It does not count towards the snapshot owner's fast snapshot restore quota.
- Amazon EBS emits Amazon CloudWatch events when the fast snapshot restore state for a snapshot changes. For more information, see EBS fast snapshot restore events.

Pricing and Billing

You are billed for each minute that fast snapshot restore is enabled for a snapshot in a particular Availability Zone. Charges are pro-rated with a minimum of one hour.

For example, if you enable fast snapshot restore for one snapshot in `US-East-1a` for one month (30 days), you are billed **$540** (1 snapshot x 1 AZ x `720` hours x `$0.75` per hour). If you enable fast snapshot restore for two snapshots in `us-east-1a`, `us-east-1b`, and `us-east-1c` for the same period, you are billed **$3240** (2 snapshots x 3 AZs x `720` hours x `$0.75` per hour).

If you enable fast snapshot restore for a public or private snapshot that is shared with you, your account is billed; the snapshot owner is not billed. When a snapshot that is shared with you is deleted or unshared by the snapshot owner, fast snapshot restore is disabled for the snapshot in your account and billing is stopped.

For more information, see Amazon EBS pricing.

Amazon EBS fast snapshot restore volume creation credits

The number of volumes that receive the full performance benefit of fast snapshot restore is determined by the volume creation credits for the snapshot. There is one credit bucket per snapshot per Availability Zone. Each volume that you create from a snapshot with fast snapshot restore enabled consumes one credit from the credit bucket. You must have at least one credit in the bucket to create an initialized volume from the snapshot. If you create a volume but there is less than one credit in the bucket, the volume is created without benefit of fast snapshot restore.

When you enable fast snapshot restore for a snapshot that is shared with you, you get a separate credit bucket for the shared snapshot in your account. If you create volumes from the shared snapshot, the credits are consumed from your credit bucket; they are not consumed from the snapshot owner's credit bucket.

The credit bucket size and the refill rate are based on the size of the snapshot (which is also the size of the source volume), not the size of the snapshot data. For example, if you create a snapshot from a 200 GiB volume that has 150 GiB of data, and enable it for fast snapshot restore, the credit bucket size and the refill rate are based on 200 GiB.

When you enable fast snapshot restore for a snapshot, the credit bucket starts with zero credits, and it gets filled at a set rate until it reaches its maximum credit capacity. Also, as you consume credits, the credit bucket is refilled over time until it reaches its maximum credit capacity.

The fill rate for a credit bucket is calculated as follows:

```
MIN (10, (1024 ÷ snapshot_size_gib))
```

And the size of the credit bucket is calculated as follows:

```
MAX (1, MIN (10, (1024 ÷ snapshot_size_gib)))
```

For example, if you enable fast snapshot restore for a snapshot with a size of `128 GiB`, the fill rate is `0.1333` credits per minute.

```
MIN (10, (1024 ÷ 128))
 = MIN (10, 8)
 = 8 credits per hour
 = 0.1333 credits per minute
```

And the maximum size of the credit bucket is 8 credits.

```
MAX (1, MIN (10, (1024 ÷ 128)))
 = MAX (1, MIN (10, 8))
 = MAX (1, 8)
 = 8 credits
```

In this example, when you enable fast snapshot restore, the credit bucket starts with zero credits. After 8 minutes, the credit bucket has enough credits to create one initialized volume (`0.1333 credits × 8 minutes = 1.066 credits`). When the credit bucket is full, you can create 8 initialized volumes simultaneously (8 credits). When the bucket is below its maximum capacity, it refills with `0.1333` credits per minute.

You can use CloudWatch metrics to monitor the size of your credit buckets and the number of credits available in each bucket. For more information, see [Metrics for fast snapshot restore](#).

After you create a volume from a snapshot with fast snapshot restore enabled, you can describe the volume using [describe-volumes](#) and check the `fastRestored` field in the output to determine whether the volume was created as an initialized volume using fast snapshot restore.

Configure fast snapshot restore for an Amazon EBS snapshot

Fast snapshot restore is disabled for a snapshot by default. You can enable or disable fast snapshot restore for snapshots that you own and for snapshots that are shared with you. When you enable or disable fast snapshot restore for a snapshot, the changes apply to your account only.

> **Note**
>
> When you enable fast snapshot restore for a snapshot, your account is billed for each minute that fast snapshot restore is enabled in a particular Availability Zone. Charges are pro-rated and have a minimum of one hour.

When you delete a snapshot that you own, fast snapshot restore is automatically disabled for that snapshot in your account. If you enabled fast snapshot restore for a snapshot that is shared with you, and the snapshot owner deletes or unshares it, fast snapshot restore is automatically disabled for the shared snapshot in your account.

If you enabled fast snapshot restore for a snapshot that is shared with you, and it has been encrypted using a custom CMK, fast snapshot restore is not automatically disabled for the snapshot when the snapshot owner revokes your access to the custom CMK. You must manually disable fast snapshot restore for that snapshot.

After you enable fast snapshot restore for a snapshot, it enters the `optimizing` state. Snapshots that are in the `optimizing` state provide some performance benefits when using them to restore volumes. They start to provide the full performance benefits of fast snapshot restore only after they enter the `enabled` state.

Console

To configure fast snapshot restore

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot, and choose **Actions**, **Manage fast snapshot restore**.
4. The **Fast snapshot restore settings** section lists all of the Availability Zones in which you can enable fast snapshot restore for the selected snapshot. The **Current status** volume indicates whether fast snapshot restore is current enabled or disabled for each zone.

   To enable fast snapshot restore in a zone where it is currently disabled, select the zone, choose **Enable**, and then to confirm, choose **Enable**.

   To disable fast snapshot restore in a zone where it is currently enabled, select the zone, and then choose **Disable**.
5. After you have made the required changes, choose **Close**.

AWS CLI

To enable fast snapshot restore

Use the enable-fast-snapshot-restores command.

```
aws ec2 enable-fast-snapshot-restores \
    --availability-zones us-east-1a us-east-1b \
    --source-snapshot-ids snap-0abcdef1234567890
```

To disable fast snapshot restore

Use the disable-fast-snapshot-restores command.

```
aws ec2 disable-fast-snapshot-restores \
    --availability-zones us-east-1a \
    --source-snapshot-ids snap-0abcdef1234567890
```

The following example uses the describe-fast-snapshot-restores command to describe your disabled fast snapshot restores.

```
aws ec2 describe-fast-snapshot-restores \
    --filters Name=state,Values=disabled
```

PowerShell

To enable fast snapshot restore

Use the Enable-EC2FastSnapshotRestore cmdlet.

```
Enable-EC2FastSnapshotRestore `
    -AvailabilityZone us-east-1a us-east-1b `
    -SourceSnapshotId snap-0abcdef1234567890
```

To disable fast snapshot restore

Use the Disable-EC2FastSnapshotRestore cmdlet.

```
Disable-EC2FastSnapshotRestore `
    -AvailabilityZone us-east-1a `
    -SourceSnapshotId snap-0abcdef1234567890
```

The following example uses the Get-EC2FastSnapshotRestore cmdlet to describe your disabled fast snapshot restores.

```
Get-EC2FastSnapshotRestore `
    -Filter @{Name="state"; Values="disabled"}
```

Check the fast snapshot restore state for an Amazon EBS snapshot

Fast snapshot restore for a snapshot can be in one of the following states.

- `enabling` — A request was made to enable fast snapshot restore.

- `optimizing` — Fast snapshot restore is being enabled. It takes 60 minutes per TiB to optimize a snapshot. Snapshots in this state offer some performance benefit when restoring volumes.
- `enabled` — Fast snapshot restore is enabled. Snapshots that are in this state and that have sufficient volume creation credits offer the full performance benefit when restoring volumes.
- `disabling` — A request was made to disable fast snapshot restore, or a request to enable fast snapshot restore failed.
- `disabled` — Fast snapshot restore is disabled. You can enable fast snapshot restore again as needed.

You can view the state of fast snapshot restore for a snapshot that you own or for a snapshot that is shared with you.

Console

To view the state of fast snapshot restore

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot.
4. On the **Details** tab, **Fast snapshot restore**, indicates the state of fast snapshot restore.

AWS CLI

To view snapshots with fast snapshot restore enabled

Use the describe-fast-snapshot-restores command.

```
aws ec2 describe-fast-snapshot-restores --filters Name=state,Values=enabled
```

The following is example output.

```
{
    "FastSnapshotRestores": [
        {
            "SnapshotId": "snap-0e946653493cb0447",
            "AvailabilityZone": "us-east-2a",
            "State": "enabled",
```

```
            "StateTransitionReason": "Client.UserInitiated - Lifecycle state
 transition",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.596Z",
            "OptimizingTime": "2020-01-25T23:58:25.573Z",
            "EnabledTime": "2020-01-25T23:59:29.852Z"
        },
        {
            "SnapshotId": "snap-0e946653493cb0447",
            "AvailabilityZone": "us-east-2b",
            "State": "enabled",
            "StateTransitionReason": "Client.UserInitiated - Lifecycle state
 transition",
            "OwnerId": "123456789012",
            "EnablingTime": "2020-01-25T23:57:49.596Z",
            "OptimizingTime": "2020-01-25T23:58:25.573Z",
            "EnabledTime": "2020-01-25T23:59:29.852Z"
        }
    ]
}
```

PowerShell

To view snapshots with fast snapshot restore enabled

Use the Get-EC2FastSnapshotRestore cmdlet.

```
Get-EC2FastSnapshotRestore `
    -Filter @{Name="state"; Values="enabled"}
```

View Amazon EBS volumes restored using fast snapshot restore

When you create a volume from a snapshot that is enabled for fast snapshot restore in the Availability Zone for the volume, it is restored using fast snapshot restore.

AWS CLI

To view volumes that were created from a snapshot that is enabled for fast snapshot restore

Use the describe-volumes command.

```
aws ec2 describe-volumes --filters Name=fast-restored,Values=true
```

The following is example output.

```
{
    "Volumes": [
        {
            "Attachments": [],
            "AvailabilityZone": "us-east-2a",
            "CreateTime": "2020-01-26T00:34:11.093Z",
            "Encrypted": true,
            "KmsKeyId": "arn:aws:kms:us-west-2:123456789012:key/8c5b2c63-b9bc-45a3-
a87a-5513e232e843",
            "Size": 20,
            "SnapshotId": "snap-0abcdef1234567890",
            "State": "available",
            "VolumeId": "vol-01234567890abcdef",
            "Iops": 100,
            "VolumeType": "gp2",
            "FastRestored": true
        }
    ]
}
```

PowerShell

To view volumes that were created from a snapshot that is enabled for fast snapshot restore

Use the Get-EC2Volume cmdlet.

```
Get-EC2Volume -Filter @{Name="fast-restored"; Values="true"}
```

Amazon EBS snapshot lock

You can lock your Amazon EBS snapshots to protect them against accidental or malicious deletions, or to store them in WORM (write-once-read-many) format for a specific duration. While a snapshot is locked, it can't be deleted by any user, regardless of their IAM permissions. You can continue to use a locked snapshot in the same way that you would use any other snapshot.

> **Note**
>
> Snapshot lock has been assessed by Cohasset Associates for use in environments that are subject to SEC 17a-4, CFTC, and FINRA regulations. For more information about how

snapshot lock relates to these regulations, see the Cohasset Associates Compliance Assessment.

You can lock snapshots in one of two modes: *compliance mode* or *governance mode*, and they can be locked for a specific duration or until a specific date. For more information, see Lock mode and Lock duration.

Pricing

You can lock and unlock snapshots at no additional cost. You pay the standard Amazon EBS snapshot storage costs for locked snapshots.

Topics

Amazon EBS snapshot lock concepts

The following are important concepts to understand as you get started using snapshot lock.

Contents

Lock mode

You can lock a snapshot in one of two modes:

Governance mode

After a snapshot is locked, users with appropriate IAM permissions can unlock the snapshot and modify the lock mode and lock duration or expiry date at any time. When you lock a snapshot in governance mode, the snapshot is locked immediately; there is no cooling-off period. To delete a snapshot after it has been locked in governance mode, you must first unlock the snapshot or you must wait for the lock to expire.

You can use governance mode to meet your organization's data governance requirements by ensuring that only certain users have permission to unlock snapshots and modify snapshot lock configurations. You can also use governance mode to test your lock configuration before locking a snapshot in compliance mode.

Compliance mode

When you lock a snapshot in compliance mode, you can optionally specify a cooling-off period that starts immediately after you lock the snapshot. During the cooling-off period, users with appropriate permissions can unlock the snapshot, change the lock mode, increase or decrease the cooling-off period, and increase or decrease the lock duration or expiry date. After the cooling-off period expires, you can't unlock the snapshot, change the lock mode, or decrease the lock duration or expire date; you can only increase the lock duration or expiry date. To delete a snapshot after it has been locked in compliance and the cooling-off period has expired, you must wait for the lock to expire.

> **Note**
>
> You can lock a snapshot in compliance mode without a cooling-off period by omitting the cooling-off period in the request. If you do this, the lock becomes effective immediately, and you can't unlock the snapshot, change the lock mode, or decrease the lock duration or expire date; you can only increase the lock duration or expiry date.

You can use compliance mode to protect snapshots that should not be deleted for a specific period for compliance reasons. Compliance mode offers the following benefits:

- It enables WORM (write-once, read-many) configuration for your snapshots.
- It provides an additional layer of defense that protects snapshots from accidental or malicious deletions.

- It enforces retention periods, which prevent early deletions by privileged users, to meet your organization's data protection policies and procedures.

> **Note**
>
> The only way to delete a snapshot that is locked in compliance mode before its lock expires is to close the associated AWS account.

Lock duration

The lock duration is the period of time for which the snapshot is to remain locked. You can specify the lock duration as one of the following, but not both:

Number of days

The lock duration is specified as a number of days for which the snapshot is to remain locked. After the specified number of days has passed, the snapshot is automatically unlocked. The duration can range from 1 day to 36500 days (100 years).

Lock expiration date

The lock duration is determined by an expiration date in the future. The snapshot remains locked until the lock expiration date is reached. When the lock expiration date is reached, the snapshot is automatically unlocked.

Cooling-off period

The cooling-off period is an optional period of time that you can specify when you lock a snapshot in compliance mode. During the cooling-off period, users with appropriate permissions can unlock the snapshot, change the lock mode, increase or decrease the cooling-off period, and increase or decrease the lock duration. After the cooling-off period expires, users can't unlock the snapshot, change the lock mode, reinstate the cooling-off period, or decrease the lock duration, regardless of their permissions.

A snapshot can't be deleted during the cooling-off period.

If specified, the cooling-off period starts immediately after you lock the snapshot. If omitted, the snapshot is locked in compliance mode immediately without a cooling-off period.

The cooling-off period can range from 1 to 72 hours. To lock a snapshot in compliance mode immediately without a cooling-off period, do not specify a cooling-off period in the request.

Lock state

A snapshot lock can be in one of the following states:

- `compliance-cooloff` — The snapshot has been locked in compliance mode but it is still within the cooling-off period. The snapshot can't be deleted, but it can be unlocked and the lock settings can be modified by users with appropriate permissions.
- `governance` — The snapshot is locked in governance mode. The snapshot can't be deleted, but it can be unlocked and the lock settings can be modified by users with appropriate permissions.
- `compliance` — The snapshot is locked in compliance mode without a cooling-off period or the cooling-off period has expired. The snapshot can't be unlocked or deleted. The lock duration can only be increased by users with appropriate permissions.
- `expired` — The snapshot was locked in compliance or governance mode but the lock has expired. The snapshot is not locked and can be deleted.

Considerations for Amazon EBS snapshot lock

Keep the following in mind when locking Amazon EBS snapshots.

- You can lock a snapshot only if it is in the `pending` or `completed` state.
  - If you lock a snapshot while it is in the `pending` state, and you lock it for a specific duration, the lock duration starts only when the snapshot reaches the `completed` state. The snapshot can't be deleted while it is in the `pending` state.
  - If you lock a snapshot while it is in the `pending` state and the snapshot creation fails for any reason, the lock is canceled.
- If you extend the lock duration for a snapshot that is locked in compliance mode after the cooling-off period has expired, you can't specify another cooling-off period. If you specify a cooling-off period, the request fails.
- You can lock archived snapshots. And you can archive locked snapshots.
- You can lock snapshots that are associated with an AMI.
- You can deregister an AMI that has associated snapshots that are locked.
- You can delete the KMS key used to encrypt a locked snapshot.

- We recommend that you do not lock snapshots created by AWS Backup. AWS Backup already ensures that its snapshots are not deleted before their retention period expires. To add an additional layer of security for snapshots managed by AWS Backup, we recommend that you use AWS Backup Vault Lock. For more information, see AWS Backup Vault Lock.
- You can't lock snapshots during creation or during AMI registration.
- You can't lock local Amazon EBS snapshots on AWS Outposts.
- The only way to delete a snapshot that is locked in compliance mode before its lock expires is to close the associated AWS account.

  If you close your AWS account while you have locked snapshots, AWS suspends your account for 90 days with your snapshots intact. If you do not reopen your account within the 90 days, AWS deletes your snapshots, even if they are locked.

Control access to Amazon EBS snapshot lock

By default, users don't have permission to work with snapshot locks. To allow users to use snapshot locks, you must create IAM policies that grant permission to use specific resources and API actions. For more information, see Creating IAM policies in the IAM User Guide.

Topics

Required permissions

To work with snapshot locks, users need the following permissions.

- `ec2:LockSnapshot` — To lock snapshots.
- `ec2:UnlockSnapshot` — To unlock snapshots.
- `ec2:DescribeLockedSnapshots` — To view snapshot lock settings.

The following is an example IAM policy that gives users permission to lock and unlock snapshots, and to view snapshot lock settings. It includes the `ec2:DescribeSnapshots` permission for console users. If some permissions are not needed, you can remove them from the policy.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSnapshotLockOperations",
      "Effect": "Allow",
      "Action": [
        "ec2:LockSnapshot",
        "ec2:UnlockSnapshot",
        "ec2:DescribeLockedSnapshots",
        "ec2:DescribeSnapshots"
      ],
      "Resource": [
        "arn:aws:ec2:*::snapshot/*",
        "arn:aws:ec2:*:111122223333:volume/*"
      ]
    }
  ]
}
```

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Restrict access with condition keys

You can use condition keys to restrict how users are allowed to lock snapshots.

Topics

- ec2:SnapshotLockDuration
- ec2:CoolOffPeriod

ec2:SnapshotLockDuration

You can use the `ec2:SnapshotLockDuration` condition key to restrict users to specific lock durations when locking snapshots.

The following example policy restricts users to specifying a lock duration between `10` and `50` days.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSnapshotLockWithDurationCondition",
      "Effect": "Allow",
      "Action": "ec2:LockSnapshot",
      "Resource": "arn:aws:ec2:*::snapshot/*",
      "Condition": {
        "NumericGreaterThan": {
          "ec2:SnapshotLockDuration": 10
        },
        "NumericLessThan": {
          "ec2:SnapshotLockDuration": 50
        }
      }
    }
  ]
}
```

ec2:CoolOffPeriod

You can use the `ec2:CoolOffPeriod` condition key to prevent users from locking snapshots in compliance mode without a cooling-off period.

The following example policy restricts users to specifying a cooling-off period greater than 48 hours when locking snapshots in compliance mode.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowSnapshotLockWithCondition",
      "Effect": "Allow",
      "Action": "ec2:LockSnapshot",
      "Resource": "arn:aws:ec2:*::snapshot/*",
      "Condition": {
        "NumericGreaterThan": {
          "ec2:SnapshotTime": 48
        }
      }
    }
  ]
}
```

Lock an Amazon EBS snapshot

You can lock a snapshot that is in the `pending` or `completed` state. For more information, see Considerations for Amazon EBS snapshot lock.

Console

To lock a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.

3. Select the snapshot to lock and choose **Actions**, **Snapshot settings**, **Manage snapshot lock**.
4. Select **Lock snapshot**.
5. For **Lock mode**, choose either **Governance mode** or **Compliance mode**. For more information, see Lock mode.
6. For **Lock duration**, do one of the following:

   - To lock the snapshot for a specific period, choose **Lock snapshot for**, and then enter the period in either days or years.
   - To lock the snapshot until a specific date and time, choose **Lock snapshot until**, and then select the expiration date and time.

   For more information, see Lock duration.
7. (*Compliance mode only*) For **Cooling-off period**, specify a cooling-off period during which you can unlock the snapshot and modify the lock configuration. For more information, see Cooling-off period.
8. (*Compliance mode only*) To confirm that you want to lock the snapshot in compliance mode and that you will not be able to unlock the snapshot after the cooling-off period expires, choose **Acknowledge**.
9. Choose **Save lock settings**.

AWS CLI

To lock a snapshot in governance mode

Use the lock-snapshot command. For `--lock-mode`, specify `governance`. To lock the snapshot for a specific period, for `--lock-duration`, specify the period, in days.

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0abcdef1234567890 \
    --lock-mode governance \
    --lock-duration 30
```

To lock the snapshot until a specific date, for `--expiration-date`, specify the date and time at which the lock must expire, in the UTC time zone.

```
aws ec2 lock-snapshot \
```

```
    --snapshot-id snap-0abcdef1234567890 \
    --lock-mode governance \
    --expiration-date YYYY-MM-DDThh:mm:ss.sssZ
```

To lock a snapshot in compliance mode

Use the lock-snapshot command. For `--lock-mode`, specify `compliance`. For `--cool-off-period`, optionally specify a cooling-off period, in hours. To lock the snapshot for a specific period, for `--lock-duration`, specify the number of days to lock the snapshot.

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0abcdef1234567890 \
    --lock-mode compliance \
    --cool-off-period 24 \
    --lock-duration 30
```

To lock the snapshot until a specific date, for `--expiration-date`, specify the date and time at which the lock must expire, in the UTC time zone.

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0abcdef1234567890 \
    --lock-mode compliance \
    --expiration-date YYYY-MM-DDThh:mm:ss.sssZ
```

PowerShell

To lock a snapshot in governance mode

Use the Lock-EC2Snapshot cmdlet. You can optionally specify the duration of the snapshot lock, in days.

```
Lock-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890 `
    -LockMode "governance" `
    -LockDuration 30
```

Alternatively, you can lock the snapshot until a specific date, in the UTC time zone.

```
Lock-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890 `
    -LockMode "governance" `
```

```
    -ExpirationDate YYYY-MM-DDThh:mm:ss.sssZ
```

To lock a snapshot in compliance mode

Use the Lock-EC2Snapshot cmdlet. You can optionally specify a cooling-off period, in hours. You can also optionally specify the duration of the snapshot lock, in days.

```
Lock-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890 `
    -LockMode "compliance" `
    -CoolOffPeriod 24 `
    -LockDuration 30
```

Alternatively, you can lock the snapshot until a specific date, in the UTC time zone.

```
Lock-EC2Snapshot `
    -SnapshotId snap-0abcdef1234567890 `
    -LockMode "compliance" `
    -ExpirationDate YYYY-MM-DDThh:mm:ss.sssZ
```

Unlock an Amazon EBS snapshot

You can unlock a snapshot only if it is locked in governance mode, or if it is locked in compliance mode and it is still within the cooling-off period.

Console

To unlock a snapshot

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot to unlock and choose **Actions**, **Snapshot settings**, **Manage snapshot lock**.
4. Choose **Unlock snapshot** and then choose **Unlock snapshot** again to confirm.

AWS CLI

To unlock a snapshot

Use the unlock-snapshot command.

```
aws ec2 unlock-snapshot --snapshot-id snap-0abcdef1234567890
```

PowerShell

To unlock a snapshot

Use the Unlock-EC2Snapshot cmdlet.

```
Unlock-EC2Snapshot -SnapshotId snap-0abcdef1234567890
```

Update Amazon EBS snapshot lock settings

The allowed updates depend on the lock state:

- `governance` — you can change the lock mode and increase or decrease the lock duration or expiration date.
- `compliance-cooloff` — you can change the lock mode, increase or decrease the cooling-off period, and increase or decrease the lock duration or expiration date.
- `compliance` — you can only increase the lock duration or expiration date.

Console

To update snapshot lock settings

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Snapshots**.
3. Select the snapshot for which to modify the lock settings and choose **Actions**, **Snapshot settings**, **Manage snapshot lock**.
4. Update the settings as needed, and then choose **Save lock settings**.

AWS CLI

To update snapshot lock settings

Use the lock-snapshot command. Specify the ID of the snapshot and the options to modify. The following example changes the expiration date.

```
aws ec2 lock-snapshot \
    --snapshot-id snap-0abcdef1234567890 \
    --lock-mode governance \
    --expiration-date YYYY-MM-DDThh:mm:ss.sssZ
```

PowerShell

To update snapshot lock settings

Use the Lock-EC2Snapshot cmdlet. Specify the ID of the snapshot and the options to modify. The following example changes the expiration date.

```
Lock-EC2Snapshot `
    -SnapshoId snap-0abcdef1234567890 `
    -LockMode "governance" `
    -ExpirationDate YYYY-MM-DDThh:mm:ss.sssZ
```

Monitor Amazon EBS snapshot lock

You can monitor actions related to Amazon EBS snapshot lock using the following tools:

Topics

Monitor Amazon EBS snapshot locks using AWS CloudTrail

You can monitor API calls for snapshot locks as events, including calls from the console and from code calls to the APIs. Using the information collected by CloudTrail, you can determine the request that was made, the IP address from which the request was made, who made the request, when it was made, and additional details.

For more information, see Log API calls using AWS CloudTrail.

Monitor Amazon EBS snapshot locks using Amazon EventBridge

Amazon EBS emits events related to snapshot lock actions. You can use AWS Lambda and Amazon EventBridge to handle event notifications programmatically. Events are emitted on a best effort basis. For more information, see the Amazon EventBridge User Guide.

The following events are emitted:

- Successfully locked snapshot in governance or compliance mode.

```
{
  "version": "0",
  "id": "01234567-01234-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "lockSnapshot",
    "result": "succeeded",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef",
    "source": 012345678901,
    "lockState": "compliance-cooloff",
    "lockCreatedOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockExpiresOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockDuration": 123,
    "lockStartDurationTime": "yyyy-mm-ddThh:mm:ssZ",
    "cooOffPeriod": 24,
    "coolOffPeriodExpiresOn": "yyyy-mm-ddThh:mm:ssZ"
  }
}
```

- Failed lock event when a snapshot is locked while it is in the `pending` state, and it fails to reach the `completed` state.

```
{
  "version": "0",
  "id": "01234567-01234-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef"
```

```
  ],
  "detail": {
    "event": "lockSnapshot",
    "result": "failed",
    "cause": "snapshot failed",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef",
    "lockState": "pending-compliance",
    "lockCreatedOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockDuration": 123,
    "lockStartDurationTime": "yyyy-mm-ddThh:mm:ssZ",
    "cooOffPeriod": 24,
    "coolOffPeriodExpiresOn": "yyyy-mm-ddThh:mm:ssZ"
  }
}
```

- Lock expired

```
{
  "version": "0",
  "id": "01234567-01234-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "lockDurationExpiry",
    "result": "succeeded",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef",
    "lockState": "expired",
    "lockCreatedOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockExpiresOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockDuration": 123
  }
}
```

- Cooling-off period expired after being locked in compliance mode.

```
{
  "version": "0",
```

```
  "id": "01234567-01234-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef"
  ],
  "detail": {
    "event": "cooloffperiodExpiry",
    "result": "succeeded",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567890abcdef",
    "lockState": "compliance",
    "lockCreatedOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockExpiresOn": "yyyy-mm-ddThh:mm:ssZ",
    "lockDuration": 123,
    "lockStartDurationTime": "yyyy-mm-ddThh:mm:ssZ",
    "cooOffPeriod": 24,
    "coolOffPeriodExpiresOn": "yyyy-mm-ddThh:mm:ssZ"
  }
}
```

Block public access for Amazon EBS snapshots

To prevent public sharing of your snapshots, you can enable *block public access for snapshots*. After you enable block public access for snapshots in a Region, any attempt to publicly share snapshots in that Region is automatically blocked. This can help you to improve the security of your snapshots and to protect your snapshot data from unauthorized or unintended access.

Block public access for snapshots can be enabled in one of two modes:

- **Block all sharing** — Blocks all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.
- **Block new sharing** — Blocks only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.

Considerations

Keep the following in mind when working with block public access for snapshots.

- Block public access for snapshots does not prevent private snapshot sharing.
- Enabling block public access for snapshots in *block all sharing* mode does not change the permissions for snapshots that are already publicly shared. Instead, it prevents these snapshots from be publicly visible and publicly accessible. Therefore, the attributes for these snapshots still indicate that they are publicly shared, even though they are not publicly available.

  If you later disable block public access or change the mode to *block new sharing*, these snapshots will become publicly available again.
- Block public access for snapshots is a Regional setting. It applies to all snapshots in the Region in which it is enabled. You need to enable block public access for snapshots in each Region in which you want to prevent the public sharing of your snapshots.
- Block public access is an account-level setting. It applies to all users, including administrator users, in the account. You can't enable block public access for snapshots at the organization level.
- The block public access setting is configured either directly in the account or by using a declarative policy. Using a declarative policy allows you to apply the setting across multiple Regions simultaneously, as well as across multiple accounts simultaneously. When a declarative policy is in use, you can't modify the setting directly within an account. This topic describes how to configure the setting directly within an account. For information about using declarative policies, see Declarative policies in the *AWS Organizations User Guide*.
- Block public access for snapshots does not prevent the public sharing of EBS-backed AMIs. If you enable block public access for snapshots, users can still publicly share EBS-backed AMIs. If an EBS-backed AMI is publicly shared, users with access to that AMI can create volumes from its associated snapshots. To prevent public sharing of your AMIs, enable *block public access for AMIs*.
- Block public access for snapshots is not supported with local snapshots on AWS Outposts.

Pricing

Block public access for snapshots can be enabled at no additional cost.

Contents

- Configure block public access for Amazon EBS snapshots
- View the block public access setting for Amazon EBS snapshots
- Disable block public access for Amazon EBS snapshots
- Monitor block public access for Amazon EBS snapshots using EventBridge

IAM permissions for block public access for Amazon EBS snapshots

By default, users don't have permission to work with block public access for snapshots. To allow users to work with block public access for snapshots, you must create IAM policies that grant permission to use specific API actions. Once the policies are created, you must add permissions to your users, groups, or roles.

To work with block public access for snapshots, users need the following permissions.

- `ec2:EnableSnapshotBlockPublicAccess` — Enable block public access for snapshots and modify the mode.
- `ec2:DisableSnapshotBlockPublicAccess` — Disable block public access for snapshots.
- `ec2:GetSnapshotBlockPublicAccessState` — View the block public access for snapshots setting for a Region.

The following is an example IAM policy. If some permissions are not needed, you can remove them from the policy.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [{
        "Effect": "Allow",
        "Action": [
            "ec2:EnableSnapshotBlockPublicAccess",
            "ec2:DisableSnapshotBlockPublicAccess",
            "ec2:GetSnapshotBlockPublicAccessState"
        ],
        "Resource": "*"
    }]
}
```

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Configure block public access for Amazon EBS snapshots

Enable block public access for snapshots to prevent the public sharing of snapshots in the Region. After this feature is enabled, requests to publicly share snapshots in the Region are blocked.

> **Important**
>
> Enabling block public access for snapshots in *block all sharing* mode does not change the permissions for snapshots that are already publicly shared. Instead, it prevents these snapshots from be publicly visible and publicly accessible. Therefore, the attributes for these snapshots still indicate that they are publicly shared, even though they are not publicly available.
> If you later disable block public access or change the mode to *block new sharing*, these snapshots will become publicly available again.

> **Note**
>
> This setting is configured at the account level, either directly in the account or by using a declarative policy. It must be configured in each AWS Region where you want to prevent the public sharing of snapshots. Using a declarative policy allows you to apply

the setting across multiple Regions simultaneously, as well as across multiple accounts simultaneously. When a declarative policy is in use, you can't modify the setting directly within an account. This topic describes how to configure the setting directly within an account. For information about using declarative policies, see Declarative policies in the *AWS Organizations User Guide*.

Console

To configure block public access for snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **EC2 Dashboard**, and then in **Account attributes** (on the right-hand side), choose **Data protection and security**.
3. In the **Block public access for EBS snapshots** section, choose **Manage**.
4. Select **Block public access** and then choose one of the following options:
   - **Block all public access** — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.
   - **Block new public sharing** — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.
5. Choose **Update**.

AWS CLI

To enable or modify block public access for snapshots

Use the enable-snapshot-block-public-access command. For `--state` specify one of the following values:

- `block-all-sharing` — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.

- `block-new-sharing` — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.

To enable or modify block public access for snapshots for a specific Region

```
aws ec2 enable-snapshot-block-public-access \
    --state block-new-sharing \
    --region us-east-1
```

The following is example output.

```
{
    "State": "block-new-sharing"
}
```

To enable or modify block public access for snapshots for all Regions

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 enable-snapshot-block-public-access \
            --region $region \
            --state block-new-sharing \
            --output text)
        echo -e "$region \t $output"
    );
done
```

The following is example output.

```
Region           Public Access State
--------------   ----------------------
ap-south-1       block-new-sharing
```

```
eu-north-1        block-new-sharing
eu-west-3         block-new-sharing
...
```

PowerShell

To enable or modify block public access for snapshots

Use the [Enable-EC2SnapshotBlockPublicAccess](#) command. For `-State` specify one of the following values:

- `block-all-sharing` — To block all public sharing of your snapshots. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are no longer publicly available.
- `block-new-sharing` — To block only new public sharing of your snapshots. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.

To enable or modify block public access for snapshots for a specific Region

```
Enable-EC2SnapshotBlockPublicAccess `
    -Region us-east-1 `
    -State block-new-sharing
```

The following is example output.

```
Value
-----
block-new-sharing
```

To enable or modify block public access for snapshots for all Regions

```
(Get-EC2Region -Region us-east-1).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region            = $_
        PublicAccessState = (
            Enable-EC2SnapshotBlockPublicAccess `
                -Region $_ `
```

```
                -State block-new-sharing)
    }
} | Format-Table -AutoSize
```

The following is example output.

```
Region          PublicAccessState
------          -----------------
ap-south-1      block-new-sharing
eu-north-1      block-new-sharing
eu-west-3       block-new-sharing
...
```

View the block public access setting for Amazon EBS snapshots

Block public access can be in one of the following states for each Region in your account.

- **Block all sharing** — All public sharing of your snapshots is blocked. Users in the account can't request new public sharing. Additionally, snapshots that were already publicly shared are treated as private and are not publicly available.
- **Block new sharing** — Only new public sharing of your snapshots is blocked. Users in the account can't request new public sharing. However, snapshots that were already publicly shared, remain publicly available.
- **Unblocked** — Public sharing is not blocked. Users can publicly share snapshots.

Console

To view the setting for block public access for snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **EC2 Dashboard**, and then in **Account attributes** (on the right-hand side), choose **Data protection and security**.
3. The **Block public access for EBS snapshots** section shows the current setting.

AWS CLI

To view the setting for block public access for snapshots

Use the get-snapshot-block-public-access-state command.

- For a specific Region

```
aws ec2 get-snapshot-block-public-access-state
```

In this example output, the ManagedBy field indicates the entity that configured the setting and account indicates that the setting was configured directly in the account. A value of declarative-policy would mean the setting was configured by a declarative policy. For more information, see Declarative policies in the *AWS Organizations User Guide*.

```
{
    "State": "unblocked",
    "ManagedBy": "account"
}
```

- For all Regions

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 get-snapshot-block-public-access-state \
            --region $region \
            --output text)
        echo -e "$region \t $output"
    );
done
```

The following is example output.

```
Region          Public Access State
--------------  ----------------------
ap-south-1      unblocked
eu-north-1      unblocked
eu-west-3       unblocked
```

PowerShell

To view the setting for block public access for snapshots

Use the Get-EC2SnapshotBlockPublicAccessState cmdlet.

- For a specific Region

```
Get-EC2SnapshotBlockPublicAccessState -Region us-east-1
```

The following is example output.

```
Value
-----
block-new-sharing
```

- For all Regions

```
(Get-EC2Region -Region us-east-1).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region            = $_
        PublicAccessState = (Get-EC2SnapshotBlockPublicAccessState -Region $_)
    }
} | Format-Table -AutoSize
```

The following is example output.

```
Region          Public Access State
--------------  ----------------------
ap-south-1      unblocked
eu-north-1      unblocked
eu-west-3       unblocked
...
```

Disable block public access for Amazon EBS snapshots

Disable block public access for snapshots to allow public sharing of snapshots in the Region. After this feature is disabled, users can publicly share snapshots in the Region.

> **Important**
>
> Enabling block public access for snapshots in *block all sharing* mode does not change the permissions for snapshots that are already publicly shared. Instead, it prevents these snapshots from be publicly visible and publicly accessible. Therefore, the attributes for these snapshots still indicate that they are publicly shared, even though they are not publicly available.
> If disable block public access, these snapshots will become publicly available again.

> **Note**
>
> This setting is configured at the account level, either directly in the account or by using a declarative policy. It must be configured in each AWS Region where you want to allow the public sharing of snapshots. Using a declarative policy allows you to apply the setting across multiple Regions simultaneously, as well as across multiple accounts simultaneously. When a declarative policy is in use, you can't modify the setting directly within an account. This topic describes how to configure the setting directly within an account. For information about using declarative policies, see Declarative policies in the *AWS Organizations User Guide*.

Console

To disable block public access for snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **EC2 Dashboard**, and then in **Account attributes** (on the right-hand side), choose **Data protection and security**.
3. In the **Block public access for EBS snapshots** section, choose **Manage**.
4. Clear **Block public access** and choose **Update**.

AWS CLI

To disable block public access for snapshots

Use the disable-snapshot-block-public-access command.

- For a specific Region

```
aws ec2 disable-snapshot-block-public-access --region us-east-1
```

The following is example output.

```
{
    "State": "unblocked"
}
```

- For all Regions

```
echo -e "Region   \t Public Access State" ; \
echo -e "-------------- \t ----------------------" ; \
for region in $(
    aws ec2 describe-regions \
        --region us-east-1 \
        --query "Regions[*].[RegionName]" \
        --output text
    );
    do (output=$(
        aws ec2 disable-snapshot-block-public-access \
            --region $region \
            --output text)
        echo -e "$region \t $output"
    );
done
```

The following is example output.

```
Region          Public Access State
--------------  ----------------------
ap-south-1      unblocked
eu-north-1      unblocked
eu-west-3       unblocked
```

PowerShell

To disable block public access for snapshots

Use the Disable-EC2SnapshotBlockPublicAccess cmdlet.

- For a specific Region

```
Disable-EC2SnapshotBlockPublicAccess -Region us-east-1
```

The following is example output.

```
Value
-----
unblocked
```

- For all Regions

```
(Get-EC2Region -Region us-east-1).RegionName | `
    ForEach-Object {
    [PSCustomObject]@{
        Region            = $_
        PublicAccessState = (Disable-EC2SnapshotBlockPublicAccess -Region $_)
    }
} | `
Format-Table -AutoSize
```

The following is example output.

```
Region          PublicAccessState
------          -----------------
ap-south-1      unblocked
eu-north-1      unblocked
eu-west-3       unblocked
...
```

Monitor block public access for Amazon EBS snapshots using EventBridge

Amazon EBS emits events related to block public access for snapshots. You can use AWS Lambda and Amazon EventBridge to handle event notifications programmatically. Events are emitted on a best effort basis. For more information, see the Amazon EventBridge User Guide.

The following events are emitted:

- Enable block public access for snapshots in block all sharing mode

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Block Public Access Enabled",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "detail": {
    "SnapshotBlockPublicAccessState": "block-all-sharing",
    "message": "Block Public Access was successfully enabled in 'block-all-sharing'
 mode"
  }
}
```

- Enable block public access for snapshots in block new sharing mode

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Block Public Access Enabled",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
  "region": "us-east-1",
  "detail": {
    "SnapshotBlockPublicAccessState": "block-new-sharing",
    "message": "Block Public Access was successfully enabled in 'block-new-sharing'
 mode"
  }
}
```

- Disable block public access for snapshots

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Block Public Access Disabled",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "2019-05-31T21:49:54Z",
```

```
  "region": "us-east-1",
  "detail": {
    "SnapshotBlockPublicAccessState": "unblocked",
    "message": "Block Public Access was successfully disabled"
  }
}
```

Amazon EBS local snapshots on Outposts

Amazon EBS snapshots are a point-in-time copy of your EBS volumes.

By default, snapshots of EBS volumes on an AWS Outpost are stored in Amazon S3 in the Region of the Outpost. You can also use Amazon EBS local snapshots on Outposts to store snapshots of volumes on an Outpost locally in Amazon S3 on the Outpost itself. This ensures that the snapshot data resides on the Outpost, and on your premises. In addition, you can use AWS Identity and Access Management (IAM) policies and permissions to set up data residency enforcement policies to ensue that snapshot data does not leave the Outpost. This is especially useful if you reside in a country or region that is not yet served by an AWS Region and that has data residency requirements.

This topic provides information about working with Amazon EBS local snapshots on Outposts. For more information about Amazon EBS snapshots and about working with snapshots in an AWS Region, see Amazon EBS snapshots.

For more information, see AWS Outposts Family and the AWS Outposts Family Documentation.

Topics

Frequently asked questions

1. **What are local snapshots?**

   By default, Amazon EBS snapshots of volumes on an Outpost are stored in Amazon S3 in the Region of the Outpost. If the Outpost is provisioned with S3 on Outposts, you can choose to store the snapshots locally on the Outpost itself. Local snapshots are incremental, which means that only the blocks of the volume that have changed after your most recent snapshot are saved. You can use these snapshots to restore a volume on the same Outpost as the snapshot at any time. For more information about Amazon EBS snapshots, see Amazon EBS snapshots.

2. **Why should I use local snapshots?**

   Snapshots are a convenient way of backing up your data. With local snapshots, all of your snapshot data is stored locally on the Outpost. This means that it does not leave your premises. This is especially useful if you reside in a country or region that is not yet served by an AWS Region and that has residency requirements.

   Additionally, using local snapshots can help to reduce the bandwidth used for communication between the Region and the Outpost in bandwidth constrained environments.

3. **How do I enforce snapshot data residency on an Outpost?**

   You can use AWS Identity and Access Management (IAM) policies to control the permissions that principals (AWS accounts, IAM users, and IAM roles) have when working with local snapshots and to enforce data residency. You can create a policy that prevents principals from creating snapshots from Outpost volumes and instances and storing the snapshots in an AWS Region. Currently, copying snapshots and images from an Outpost to a Region is not supported. For more information, see Controlling access with IAM.

4. **Are multi-volume, crash-consistent local snapshots supported?**

   Yes, you can create multi-volume, crash-consistent local snapshots from instances on an Outpost.

5. **How do I create local snapshots?**

   You can create snapshots manually using the AWS Command Line Interface (AWS CLI) or the Amazon EC2 console. For more information see, Working with local snapshots. You can also automate the lifecycle of local snapshots using Amazon Data Lifecycle Manager. For more information see, Automate snapshots on an Outpost.

6. Can I create, use, or delete local snapshots if my Outpost loses connectivity to its Region?

No. The Outpost must have connectivity with its Region as the Region provides the access, authorization, logging, and monitoring services that are critical for your snapshots' health. If there is no connectivity, you can't create new local snapshots, create volumes or launch instances from existing local snapshots, or delete local snapshots.

7. How quickly is Amazon S3 storage capacity made available after deleting local snapshots?

Amazon S3 storage capacity becomes available within 72 hours after deleting local snapshots and the volumes that reference them.

8. How can I ensure that I do not run out of Amazon S3 capacity on my Outpost?

We recommend that you use Amazon CloudWatch alarms to monitor your Amazon S3 storage capacity, and delete snapshots and volumes that you no longer need to avoid running out of storage capacity. If you are using Amazon Data Lifecycle Manager to automate the lifecycle of local snapshots, ensure that your snapshot retention policies do not retain snapshots for longer than is needed.

9. What happens if I run out of local Amazon S3 capacity on an Outpost?

If you run out of local Amazon S3 capacity on an Outpost, Amazon Data Lifecycle Manager will not be able to successfully create local snapshots on the Outpost. Amazon Data Lifecycle Manager will attempt to create the local snapshots on the Outpost, but the snapshots immediately transition to the `error` state and they are eventually deleted by Amazon Data Lifecycle Manager. We recommend that you use the `SnapshotsCreateFailed` Amazon CloudWatch metric to monitor your snapshot lifecycle policies for snapshot creation failures. For more information, see Monitor Data Lifecycle Manager policies using CloudWatch.

10. Can I use local snapshots and AMIs backed by local snapshots with Spot Instances and Spot Fleet?

No, you can't use local snapshots or AMIs backed by local snapshots to launch Spot Instances or a Spot Fleet.

11. Can I use local snapshots and AMIs backed by local snapshots with Amazon EC2 Auto Scaling?

Yes, you can use local snapshots and AMIs backed by local snapshots to launch Auto Scaling groups in a subnet that is on the same Outpost as the snapshots. The Amazon EC2 Auto Scaling group service-linked role must have permission to use the KMS key used to encrypt the snapshots.

You can't use local snapshots or AMIs backed by local snapshots to launch Auto Scaling groups in an AWS Region.

Prerequisites

To store snapshots on an Outpost, you must have an Outpost that is provisioned with S3 on Outposts. For more information about S3 on Outposts, see S3 on Outposts in the *Amazon S3 on Outposts User Guide*.

Considerations

Keep the following in mind when working with local snapshots.

- The Outpost must have connectivity to their AWS Region to use local snapshots.
- Snapshot metadata is stored in the AWS Region associated with the Outpost. This does not include any snapshot data.
- Snapshots stored on an Outpost are encrypted by default. Unencrypted snapshots are not supported. Snapshots that are created on an Outpost and snapshots that are copied to an Outpost are encrypted using the default KMS key for the Region or a different KMS key that you specify at the time of the request.
- When you create a volume on an Outpost from a local snapshot, you cannot re-encrypt the volume using a different KMS key. Volumes created from local snapshots must be encrypted using the same KMS key as the source snapshot.
- After you delete local snapshots from an Outpost, the Amazon S3 storage capacity used by the deleted snapshots becomes available within 72 hours. For more information, see Delete local snapshots.
- You can't export local snapshots from an Outpost.
- You can't enable fast snapshot restore for local snapshots.
- EBS direct APIs are not supported with local snapshots.
- You can't copy local snapshots or AMIs from an Outpost to an AWS Region, from one Outpost to another, or within an Outpost. However, you can copy snapshots from an AWS Region to an Outpost. For more information, see Copy snapshots from an AWS Region to an Outpost.
- When copying a snapshot from an AWS Region to an Outpost, the data is transferred over the service link. Copying multiple snapshots simultaneously could impact other services running on the Outpost.

- You can't share local snapshots.
- You must use IAM policies to ensure that your data residency requirements are met. For more information, see Controlling access with IAM.
- Local snapshots are incremental backups. Only the blocks in the volume that have changed after your most recent snapshot are saved. Each local snapshot contains all of the information that is needed to restore your data (from the moment when the snapshot was taken) to a new EBS volume. For more information, see How Amazon EBS snapshots work.
- You can’t use IAM policies to enforce data residency for **CopySnapshot** and **CopyImage** actions.

Controlling access with IAM

You can use AWS Identity and Access Management (IAM) policies to control the permissions that principals (AWS accounts, IAM users, and IAM roles) have when working with local snapshots. The following are example policies that you can use to grant or deny permission to perform specific actions with local snapshots.

> **Important**
>
> Copying snapshots and images from an Outpost to a Region is currently not supported. As result, you currently can’t use IAM policies to enforce data residency for **CopySnapshot** and **CopyImage** actions.

Topics

- Enforce data residency for snapshots
- Prevent principals from deleting local snapshots

Enforce data residency for snapshots

The following example policy prevents all principals from creating snapshots from volumes and instances on Outpost `arn:aws:outposts:us-east-1:123456789012:outpost/op-1234567890abcdef` and storing the snapshot data in an AWS Region. Principals can still create local snapshots. This policy ensures that all snapshots remain on the Outpost.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:SourceOutpostArn": "arn:aws:outposts:us-east-1:123456789012:outpost/op-1234567890abcdef0"
                },
                "Null": {
                    "ec2:OutpostArn": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "*"
        }
    ]
}
```

Prevent principals from deleting local snapshots

The following example policy prevents all principals from deleting local snapshots that are stored on Outpost `arn:aws:outposts:us-east-1:123456789012:outpost/op-1234567890abcdef0`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ec2:DeleteSnapshot"
            ],
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:OutpostArn": "arn:aws:outposts:us-
east-1:123456789012:outpost/op-1234567890abcdef0"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DeleteSnapshot"
            ],
            "Resource": "*"
        }
    ]
}
```

Working with local snapshots

The following sections explain how to use local snapshots.

Topics

- Launch instances from AMIs backed by local snapshots
- Delete local snapshots
- Automate snapshots on an Outpost

Rules for storing snapshots

The following rules apply to snapshot storage:

- If the most recent snapshot of a volume is stored on an Outpost, then all successive snapshots must be stored on the same Outpost.
- If the most recent snapshot of a volume is stored in an AWS Region, then all successive snapshots must be stored in the same Region. To start creating local snapshots from that volume, do the following:

  1. Create a snapshot of the volume in the AWS Region.
  2. Copy the snapshot to the Outpost from the AWS Region.
  3. Create a new volume from the local snapshot.
  4. Attach the volume to an instance on the Outpost.

  For the new volume on the Outpost, the next snapshot can be stored on the Outpost or in the AWS Region. All successive snapshots must then be stored in that same location.
- Local snapshots, including snapshots created on an Outpost and snapshots copied to an Outpost from an AWS Region, can be used only to create volumes on the same Outpost.
- If you create a volume on an Outpost from a snapshot in a Region, then all successive snapshots of that new volume must be in the same Region.
- If you create a volume on an Outpost from a local snapshot, then all successive snapshots of that new volume must be on the same Outpost.

Create local snapshots from volumes on an Outpost

You can create local snapshots from volumes on your Outpost. You can choose to store the snapshots on the same Outpost as the source volume, or in the Region for the Outpost.

Local snapshots can be used to create volumes on the same Outpost only.

For more information, see Create Amazon EBS snapshots

Create AMIs from local snapshots

You can create Amazon Machine Images (AMIs) using a combination of local snapshots and snapshots that are stored in the Region of the Outpost. For example, if you have an Outpost in `us-east-1`, you can create an AMI with data volumes that are backed by local snapshots on that Outpost, and a root volume that is backed by a snapshot in the `us-east-1` Region.

> **Note**
>
> - You can't create AMIs that include backing snapshots stored across multiple Outposts.
> - You can’t currently create AMIs directly from instances on an Outpost using **CreateImage** API or the Amazon EC2 console for an Outpost.
> - AMIs that are backed by local snapshots can be used to launch instances on the same Outpost only.

To create an AMI on an Outpost from snapshots in a Region

1. Copy the snapshots from the Region to the Outpost. For more information, see Copy snapshots from an AWS Region to an Outpost.
2. Use the Amazon EC2 console or the register-image command to create the AMI using the snapshot copies on the Outpost. For more information, see Creating an AMI from a snapshot.

To create an AMI on an Outpost from an instance on an Outpost

1. Create snapshots from the instance on the Outpost and store the snapshots on the Outpost. For more information, see Create Amazon EBS snapshots.
2. Use the Amazon EC2 console or the register-image command to create the AMI using the local snapshots. For more information, see Creating an AMI from a snapshot.

To create an AMI in a Region from an instance on an Outpost

1. Create snapshots from the instance on the Outpost and store the snapshots in the Region. For more information, see Create local snapshots from volumes on an Outpost or Create Amazon EBS snapshots.

2. Use the Amazon EC2 console or the register-image command to create the AMI using the snapshot copies in the Region. For more information, see Creating an AMI from a snapshot.

Copy snapshots from an AWS Region to an Outpost

You can copy snapshots from an AWS Region to an Outpost. You can do this only if the snapshots are in the Region for the Outpost. If the snapshots are in a different Region, you must first copy the snapshot to the Region for the Outpost, and then copy it from that Region to the Outpost.

> **Note**
>
> You can't copy local snapshots from an Outpost to a Region, from one Outpost to another, or within the same Outpost.

For more information, see Copy an Amazon EBS snapshot.

Copy AMIs from an AWS Region to an Outpost

You can copy AMIs from an AWS Region to an Outpost. When you copy an AMI from a Region to an Outpost, all of the snapshots associated with the AMI are copied from the Region to the Outpost.

You can copy an AMI from a Region to an Outpost only if the snapshots associated with the AMI are in the Region for the Outpost. If the snapshots are in a different Region, you must first copy the AMI to the Region for the Outpost, and then copy it from that Region to the Outpost.

> **Note**
>
> You can't copy an AMI from an Outpost to a Region, from one Outpost to another, or within an Outpost.

You can copy AMIs from a Region to an Outpost using the copy-image AWS CLI command only.

Create volumes from local snapshots

You can create volumes on an Outpost from local snapshots. Volumes must be created on the same Outpost as the source snapshots. You cannot use local snapshots to create volumes in the Region for the Outpost.

When you create a volume from a local snapshot, you cannot re-encrypt the volume using different KMS key. Volumes created from local snapshots must be encrypted using the same KMS key as the source snapshot.

For more information, see Create an Amazon EBS volume.

Launch instances from AMIs backed by local snapshots

You can launch instances from AMIs that are backed by local snapshots. You must launch Instances on the same Outpost as the source AMI. For more information, see Launch an instance on your Outpost in the *AWS Outposts User Guide*.

Delete local snapshots

You can delete local snapshots from an Outpost. After you delete a snapshot from an Outpost, the Amazon S3 storage capacity used by the deleted snapshot becomes available within 72 hours after deleting the snapshot and the volumes that reference that snapshot.

Because Amazon S3 storage capacity does not become available immediately, we recommend that you use Amazon CloudWatch alarms to monitor your Amazon S3 storage capacity. Delete snapshots and volumes that you no longer need to avoid running out of storage capacity.

For more information about deleting snapshots, see Delete a snapshot.

Automate snapshots on an Outpost

You can create Amazon Data Lifecycle Manager snapshot lifecycle policies that automatically create, copy, retain, and delete snapshots of your volumes and instances on an Outpost. You can choose whether to store the snapshots in a Region or whether to store them locally on an Outpost. Additionally, you can automatically copy snapshots that are created and stored in an AWS Region to an Outpost.

The following table provides an overview of the supported features.

Resource location	Snapshot destination	Cross-region copy		Fast snapshot restore	Cross-account sharing
		To Region	To Outpost		
Region	Region	✓	✓	✓	✓
Outpost	Region	✓	✓	✓	✓

Outpost	Outpost	✗	✗	✗	✗

Considerations

- Only Amazon EBS snapshot lifecycle policies are currently supported. EBS-backed AMI policies and Cross-account sharing event policies are not supported.
- If a policy manages snapshots for volumes or instances in a Region, then snapshots are created in the same Region as the source resource.
- If a policy manages snapshots for volumes or instances on an Outpost, then snapshots can be created on the source Outpost, or in the Region for that Outpost.
- A single policy can't manage both snapshots in a Region and snapshots on an Outpost. If you need to automate snapshots in a Region and on an Outpost, you must create separate policies.
- Fast snapshot restore is not supported for snapshots created on an Outpost, or for snapshots copied to an Outpost.
- Cross-account sharing is not supported for snapshots created on an Outpost.

For more information about creating a snapshot lifecycle that manages local snapshots, see Automating snapshot lifecycles.

Local snapshots in Local Zones

Amazon EBS snapshots are a point-in-time copy of your EBS volumes.

Snapshots of EBS volumes in an AWS Local Zone can be stored in Amazon S3 in the same Local Zone or in the parent Region of that Local Zone. Storing snapshots in a Local Zone can help you meet data residency needs by ensuring that snapshot data is processed and stored in a specific country, state, or municipality. You can also set up data residency enforcement policies using IAM to ensure that snapshot data does not leave the Local Zone.

Local Zones are ideal for applications that require single-digit millisecond latency or local data processing by bringing AWS infrastructure closer to your end users and business centers. Additionally, you can meet data residency requirements for regulatory and compliance-sensitive workloads. For more information, see What is AWS Local Zones.

Local snapshots are currently supported in Local Zones that support Amazon S3. For more information, see AWS Local Zones features..

Topics

- Frequently asked questions
- Considerations
- Controlling access with IAM

Frequently asked questions

1. What are Local snapshots in Local Zones?

Local snapshots in Local Zones are snapshots that are stored in Amazon S3 in a Local Zone. By default, snapshots of Amazon EBS volumes in a Local Zone are stored in Amazon S3 in the parent Region. If the Local Zone supports Amazon S3, you can choose to store the snapshots locally in the Local Zone instead. Like snapshots in AWS Regions, Local snapshots in Local Zones are incremental, which means that only the blocks of the volume that have changed after your most recent snapshot are saved. You can use these snapshots to restore an Amazon EBS volume in the same Local Zone at any time.

2. Why should I use Local snapshots?

Use Local snapshots in Local Zones to meet data residency or data isolation requirements by ensuring that your snapshot data resides in a specific geographic location, such as a country, state, or municipality.

3. How do I enforce snapshot data residency in Local Zones?

You can use AWS Identity and Access Management (IAM) policies to control the permissions that principals (AWS accounts, IAM users, and IAM roles) have when working with Local snapshots in Local Zones and to enforce data residency. For example, you can create a policy that prevents users from creating snapshots from volumes in a Local Zones and storing those snapshots in an AWS Region. For more information, see Controlling access with IAM.

4. Are multi-volume, crash-consistent Local snapshots supported?

Yes, you can create multi-volume, crash-consistent Local snapshots in Local Zones from instances in a Local Zone.

5. How do I create Local snapshots in Local Zones?

You can create Local snapshots in Local Zones manually using the AWS CLI or the Amazon EC2 console. For more information see, Create a snapshot of an EBS volume. You can also automate the lifecycle of Local snapshots in Local Zones using Amazon Data Lifecycle Manager. For more information see, Create Amazon Data Lifecycle Manager custom policy for EBS snapshots.

6. **Can I copy Local snapshots in Local Zones?**

   Yes, you can copy snapshots from a Region to a Local Zone in the same Region, from a Local Zone to its Region, and from one Local Zone to another Local Zone in the same Local Zone group.

7. **How can I restore data from Local snapshots in Local Zones?**

   You can use Local snapshots in Local Zones to create Amazon EBS volumes in the same Local Zone only.

8. **How are Local snapshots in Local Zones encrypted?**

   Local snapshots can be unencrypted or encrypted by default. When encrypted by default, Local snapshots are encrypted using the same AWS KMS key as the source Amazon EBS volume. When you create a volume from a Local snapshot, you can't re-encrypt the volume using a different KMS key. Volumes created from Local snapshots must be encrypted using the same AWS KMS key as the source snapshot.

9. **Can I create EBS-backed AMIs using Local snapshots in Local Zones?**

   Yes, you can use Local snapshots in Local Zones when creating EBS-backed AMIs by specifying the snapshot destination as Local Zones. For more information, see Create an Amazon EBS-backed AMI.

10. **Can I share Local snapshots in Local Zones?**

    Yes, you can share Local snapshots in Local Zones with other AWS accounts that have enabled the Local Zone for use in their account.

11. **Can I create a Local snapshot of a volume and then switch to creating snapshots in the parent Region?**

    No, after you create a Local snapshot of a volume, you can't create successive snapshots of that volume in the Parent Region. Since all snapshots are incremental, if the most recent snapshot of a volume is a Local snapshot, then all successive snapshots of that volume must be Local snapshots.

Considerations

Keep the following in mind when working with Local snapshots in Local Zones.

- Local snapshots are currently supported in Local Zones that support Amazon S3.
- The following features can't be used with Local snapshots in Local Zones:
  - VM Import/Export actions
  - Fast snapshot restore
  - EBS direct APIs
  - Recycle Bin
  - Snapshot archive
- You must use IAM policies to enforce your data residency requirements. For more information, see Controlling access with IAM.
- If the most recent snapshot of a volume is a Local snapshot, then all successive snapshots must be Local snapshots. Similarly, if the most recent snapshot of a volume is stored in an AWS Region, then all successive snapshots must be stored in the same Region.

Controlling access with IAM

You can use AWS Identity and Access Management (IAM) policies to control the permissions that principals (AWS accounts, IAM users, and IAM roles) have when working with Local snapshots in Local Zones. The following are example policies that you can use to grant or deny permission to perform specific actions with Local snapshots in Local Zones.

Topics

- Enforce data residency for Local snapshots in Local Zones
- Prevent sharing of Local snapshots in Local Zones
- Prevent principals from deleting Local snapshots in Local Zones

Enforce data residency for Local snapshots in Local Zones

The following example policy restricts users to creating only Local snapshots in Local Zones from volumes and instances in a Local Zone. It prevents users from creating snapshots in a Region from volumes and instances in a Local Zone.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement":
    [
        {
            "Effect": "Deny",
            "Action":
            [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "*",
            "Condition":
            {
                "StringEquals":
                {
                    "ec2:Location": "regional",
                    "ec2:SourceAvailabilityZone": "local_zone"
                }
            }
        },
        {
            "Effect": "Deny",
            "Action":
            [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "*",
            "Condition":
            {
                "StringEquals":
                {
                    "ec2:SourceAvailabilityZone": "local_zone"
                },
                "Null":
                {
                    "ec2:Location": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action":
            [
```

```
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots"
            ],
            "Resource": "*"
        }
    ]
}
```

Prevent sharing of Local snapshots in Local Zones

The following example policy prevents all users from sharing Local snapshots in Local Zones.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenySnapshotModifyInLocalZone",
      "Effect": "Deny",
      "Action": [
        "ec2:ModifySnapshotAttribute"
      ],
      "Resource": "arn:aws:ec2:*::snapshot/*",
      "Condition": {
        "StringEquals": {
        "ec2:AvailabilityZone": "use1-atl2-az1"
        }
      }
    },
    {
      "Sid": "AllowSnapshotModifyElsewhere",
      "Effect": "Allow",
      "Action": [
        "ec2:ModifySnapshotAttribute"
      ],
      "Resource": "arn:aws:ec2:*::snapshot/*"
    }
  ]
}
```

Prevent principals from deleting Local snapshots in Local Zones

The following example policy prevents all users from deleting Local snapshots in Local Zones.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenySnapshotDeleteInLocalZone",
      "Effect": "Deny",
      "Action": [
        "ec2:DeleteSnapshot"
      ],
      "Resource": "arn:aws:ec2:*::snapshot/*",
      "Condition": {
        "StringEquals": {
        "ec2:AvailabilityZone": "use1-atl2-az1"
        }
      }
    },
    {
      "Sid": "AllowSnapshotDeleteElsewhere",
      "Effect": "Allow",
      "Action": [
        "ec2:DeleteSnapshot"
      ],
      "Resource": "arn:aws:ec2:*::snapshot/*"
    }
  ]
}
```

Amazon EBS encryption

Use Amazon EBS encryption as a straight-forward encryption solution for your Amazon EBS resources associated with your Amazon EC2 instances. With Amazon EBS encryption, you aren't required to build, maintain, and secure your own key management infrastructure. Amazon EBS encryption uses AWS KMS keys when creating encrypted volumes and snapshots.

Encryption operations occur on the servers that host EC2 instances, ensuring the security of both data-at-rest and data-in-transit between an instance and its attached EBS storage.

You can attach both encrypted and unencrypted volumes to an instance simultaneously. All Amazon EC2 instance types support Amazon EBS encryption.

Contents

How Amazon EBS encryption works

You can encrypt both the boot and data volumes of an EC2 instance.

When you create an encrypted EBS volume and attach it to a supported instance type, the following types of data are encrypted:

- Data at rest inside the volume
- All data moving between the volume and the instance
- All snapshots created from the volume
- All volumes created from those snapshots

Amazon EBS encrypts your volume with a data key using industry-standard AES-256 data encryption. The data key is generated by AWS KMS and then encrypted by AWS KMS with a AWS

KMS key prior to being stored with your volume information. Amazon EBS automatically creates a unique AWS managed key in each Region where you create Amazon EBS resources. The alias for the KMS key is `aws/ebs`. By default, Amazon EBS uses this KMS key for encryption. Alternatively, you can use a symmetric customer managed encryption key that you create. Using your own KMS key gives you more flexibility, including the ability to create, rotate, and disable KMS keys.

Amazon EC2 works with AWS KMS to encrypt and decrypt your EBS volumes in slightly different ways depending on whether the snapshot from which you create an encrypted volume is encrypted or unencrypted.

How EBS encryption works when the snapshot is encrypted

When you create an encrypted volume from an encrypted snapshot that you own, Amazon EC2 works with AWS KMS to encrypt and decrypt your EBS volumes as follows:

1. Amazon EC2 sends a GenerateDataKeyWithoutPlaintext request to AWS KMS, specifying the KMS key that you chose for volume encryption.
2. If the volume is encrypted using the same KMS key as the snapshot, AWS KMS uses the same data key as the snapshot and encrypts it under that same KMS key. If the volume is encrypted using a different KMS key, AWS KMS generates a new data key and encrypts it under the KMS key that you specified. The encrypted data key is sent to Amazon EBS to be stored with the volume metadata.
3. When you attach the encrypted volume to an instance, Amazon EC2 sends a CreateGrant request to AWS KMS so that it can decrypt the data key.
4. AWS KMS decrypts the encrypted data key and sends the decrypted data key to Amazon EC2.
5. Amazon EC2 uses the plaintext data key in the Nitro hardware to encrypt disk I/O to the volume. The plaintext data key persists in memory as long as the volume is attached to the instance.

How EBS encryption works when the snapshot is unencrypted

When you create an encrypted volume from unencrypted snapshot, Amazon EC2 works with AWS KMS to encrypt and decrypt your EBS volumes as follows:

1. Amazon EC2 sends a CreateGrant request to AWS KMS, so that it can encrypt the volume that is created from the snapshot.
2. Amazon EC2 sends a GenerateDataKeyWithoutPlaintext request to AWS KMS, specifying the KMS key that you chose for volume encryption.

3. AWS KMS generates a new data key, encrypts it under the KMS key that you chose for volume encryption, and sends the encrypted data key to Amazon EBS to be stored with the volume metadata.
4. Amazon EC2 sends a Decrypt request to AWS KMS to decrypt the encrypted data key, which it then uses to encrypt the volume data.
5. When you attach the encrypted volume to an instance, Amazon EC2 sends a CreateGrant request to AWS KMS, so that it can decrypt the data key.
6. When you attach the encrypted volume to an instance, Amazon EC2 sends a Decrypt request to AWS KMS, specifying the encrypted data key.
7. AWS KMS decrypts the encrypted data key and sends the decrypted data key to Amazon EC2.
8. Amazon EC2 uses the plaintext data key in the Nitro hardware to encrypt disk I/O to the volume. The plaintext data key persists in memory as long as the volume is attached to the instance.

For more information, see How Amazon Elastic Block Store (Amazon EBS) uses AWS KMS and Amazon EC2 example two in the *AWS Key Management Service Developer Guide*.

How unusable KMS keys affect data keys

When a KMS key becomes unusable, the effect is almost immediate (subject to eventual consistency). The key state of the KMS key changes to reflect its new condition, and all requests to use the KMS key in cryptographic operations fail.

When you perform an action that makes the KMS key unusable, there is no immediate effect on the EC2 instance or the attached EBS volumes. Amazon EC2 uses the data key, not the KMS key, to encrypt all disk I/O while the volume is attached to the instance.

However, when the encrypted EBS volume is detached from the EC2 instance, Amazon EBS removes the data key from the Nitro hardware. The next time the encrypted EBS volume is attached to an EC2 instance, the attachment fails, because Amazon EBS cannot use the KMS key to decrypt the volume's encrypted data key. To use the EBS volume again, you must make the KMS key usable again.

Tip

If you no longer want access to data stored in an EBS volume encrypted with a data key generated from a KMS key that you intend to make unusable, we recommend that you detach the EBS volume from the EC2 instance before you make the KMS key unusable.

For more information, see How unusable KMS keys affect data keys in the *AWS Key Management Service Developer Guide*.

Requirements for Amazon EBS encryption

Before you begin, verify that the following requirements are met.

Requirements

- Supported volume types
- Supported instance types
- Permissions for users
- Permissions for instances

Supported volume types

Encryption is supported by all EBS volume types. You can expect the same IOPS performance on encrypted volumes as on unencrypted volumes, with a minimal effect on latency. You can access encrypted volumes the same way that you access unencrypted volumes. Encryption and decryption are handled transparently, and they require no additional action from you or your applications.

Supported instance types

Amazon EBS encryption is available on all current generation and previous generation instance types.

Permissions for users

When you use a KMS key for EBS encryption, the KMS key policy allows any user with access to the required AWS KMS actions to use this KMS key to encrypt or decrypt EBS resources. You must grant users permission to call the following actions in order to use EBS encryption:

- `kms:CreateGrant`
- `kms:Decrypt`
- `kms:DescribeKey`
- `kms:GenerateDataKeyWithoutPlainText`
- `kms:ReEncrypt`

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to `kms:CreateGrant`. Instead, use the `kms:GrantIsForAWSResource` condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "kms:CreateGrant",
            "Resource": [
                "arn:aws:kms:us-east-2:123456789012:key/abcd1234-a123-456d-a12b-
a123b4cd56ef"
            ],
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": true
                }
            }
        }
    ]
}
```

For more information, see Allows access to the AWS account and enables IAM policies in the **Default key policy** section in the *AWS Key Management Service Developer Guide*.

Permissions for instances

When an instance attempts to interact with an encrypted AMI, volume, or snapshot, a KMS key grant is issued to the instance's identity-only role. The identity-only role is an IAM role that is used by the instance to interact with encrypted AMIs, volumes, or snapshots on your behalf.

Identity-only roles do not need to be manually created or deleted, and they have no policies associated with them. Additionally, you can't access the identity-only role credentials.

> **Note**
>
> Identity-only roles are not used by applications on your instance to access other AWS KMS encrypted resources, such as Amazon S3 objects or Dynamo DB tables. These operations are done using the credentials of an Amazon EC2 instance role, or other AWS credentials that you have configured on your instance.

Identity-only roles are subject to service control policies (SCPs), and KMS key policies. If an SCP or KMS key denies the identity-only role access to a KMS key, you may fail to launch EC2 instances with encrypted volumes, or using encrypted AMIs or snapshots.

If you are creating an SCP or key policy that denies access based on network location using the `aws:SourceIp`, `aws:VpcSourceIp`, `aws:SourceVpc`, or `aws:SourceVpce` AWS global condition keys, then you must ensure that these policy statements do not apply to instance-only roles. For example policies, see Data Perimeter Policy Examples.

Identity-only role ARNs use the following format:

```
arn:aws-partition:iam::account_id:role/aws:ec2-infrastructure/instance_id
```

When a key grant is issued to an instance, the key grant is issued to the assumed-role session specific to that instance. The grantee principal ARN uses the following format:

```
arn:aws-partition:sts::account_id:assumed-role/aws:ec2-infrastructure/instance_id
```

Enable Amazon EBS encryption by default

You can configure your AWS account to enforce the encryption of the new EBS volumes and snapshot copies that you create. For example, Amazon EBS encrypts the EBS volumes created when you launch an instance and the snapshots that you copy from an unencrypted snapshot. For examples of transitioning from unencrypted to encrypted EBS resources, see Encrypt unencrypted resources.

Encryption by default has no effect on existing EBS volumes or snapshots.

Considerations

- Encryption by default is a Region-specific setting. If you enable it for a Region, you cannot disable it for individual volumes or snapshots in that Region.
- Amazon EBS encryption by default is supported on all current generation and previous generation instance types.
- If you copy a snapshot and encrypt it to a new KMS key, a complete (non-incremental) copy is created. This results in additional storage costs.

Console

To enable encryption by default for a Region

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. From the navigation bar, select the Region.
3. From the navigation pane, select **EC2 Dashboard**.
4. In the upper-right corner of the page, choose **Account Attributes**, **Data protection and security**.
5. In the **EBS encryption** section, choose **Manage**.
6. Select **Enable**. You keep the AWS managed key with the alias `aws/ebs` created on your behalf as the default encryption key, or choose a symmetric customer managed encryption key.
7. Choose **Update EBS encryption**.

AWS CLI

To view the encryption by default setting

Use the get-ebs-encryption-by-default command.

- For a specific Region

```
aws ec2 get-ebs-encryption-by-default --region region
```

- For all Regions in your account

```
echo -e "Region      \t Encrypt \t Key"; \
echo -e "----------- \t ------- \t -------" ; \
for region in $(aws ec2 describe-regions --region us-east-1 --query "Regions[*].
[RegionName]" --output text);
do
    default=$(aws ec2 get-ebs-encryption-by-default --region $region --query
 "{Encryption_By_Default:EbsEncryptionByDefault}" --output text);
    kms_key=$(aws ec2 get-ebs-default-kms-key-id --region $region | jq
 '.KmsKeyId');
    echo -e "$region \t $default \t\t $kms_key";
done
```

To enable encryption by default

Use the enable-ebs-encryption-by-default command.

- For a specific Region

```
aws ec2 enable-ebs-encryption-by-default --region region
```

- For all Regions in your account

```
echo -e "Region      \t Encrypt \t Key"; \
echo -e "----------- \t ------- \t -------" ; \
for region in $(aws ec2 describe-regions --region us-east-1 --query "Regions[*].
[RegionName]" --output text);
do
    default=$(aws ec2 enable-ebs-encryption-by-default --region $region --query
 "{Encryption_By_Default:EbsEncryptionByDefault}" --output text);
    kms_key=$(aws ec2 get-ebs-default-kms-key-id --region $region | jq
 '.KmsKeyId');
    echo -e "$region \t $default \t\t $kms_key";
done
```

To disable encryption by default

Use the disable-ebs-encryption-by-default command.

- For a specific Region

```
aws ec2 disable-ebs-encryption-by-default --region region
```

- For all Regions in your account

```
echo -e "Region      \t Encrypt \t Key"; \
echo -e "----------- \t ------- \t -------" ; \
for region in $(aws ec2 describe-regions --region us-east-1 --query "Regions[*].
[RegionName]" --output text);
do
    default=$(aws ec2 disable-ebs-encryption-by-default --region $region --query
 "{Encryption_By_Default:EbsEncryptionByDefault}" --output text);
    kms_key=$(aws ec2 get-ebs-default-kms-key-id --region $region | jq
 '.KmsKeyId');
    echo -e "$region \t $default \t\t $kms_key";
done
```

PowerShell

To view the encryption by default setting

Use the Get-EC2EbsEncryptionByDefault cmdlet.

- For a specific Region

```
Get-EC2EbsEncryptionByDefault -Region region
```

- For all Regions in your account

```
(Get-EC2Region).RegionName |
    ForEach-Object {
    [PSCustomObject]@{
        Region                      = $_
        EC2EbsEncryptionByDefault = Get-EC2EbsEncryptionByDefault -Region $_
        EC2EbsDefaultKmsKeyId      = Get-EC2EbsDefaultKmsKeyId -Region $_
    } } |
```

```
    Format-Table -AutoSize
```

To enable encryption by default

Use the Enable-EC2EbsEncryptionByDefault cmdlet.

- For a specific Region

```
Enable-EC2EbsEncryptionByDefault -Region region
```

- For all Regions in your account

```
(Get-EC2Region).RegionName |
    ForEach-Object {
    [PSCustomObject]@{
        Region                    = $_
        EC2EbsEncryptionByDefault = Enable-EC2EbsEncryptionByDefault -Region $_
        EC2EbsDefaultKmsKeyId     = Get-EC2EbsDefaultKmsKeyId -Region $_
    } } |
    Format-Table -AutoSize
```

To disable encryption by default

Use the Disable-EC2EbsEncryptionByDefault cmdlet.

- For a specific Region

```
Disable-EC2EbsEncryptionByDefault -Region region
```

- For all Regions in your account

```
(Get-EC2Region).RegionName |
    ForEach-Object {
    [PSCustomObject]@{
        Region                    = $_
        EC2EbsEncryptionByDefault = Disable-EC2EbsEncryptionByDefault -Region $_
        EC2EbsDefaultKmsKeyId     = Get-EC2EbsDefaultKmsKeyId -Region $_
    } } |
    Format-Table -AutoSize
```

You can't change the KMS key that is associated with an existing snapshot or encrypted volume. However, you can associate a different KMS key during a snapshot copy operation so that the resulting copied snapshot is encrypted by the new KMS key.

Encrypt EBS resources

You encrypt EBS volumes by enabling encryption, either using encryption by default or by enabling encryption when you create a volume that you want to encrypt.

When you encrypt a volume, you can specify the symmetric encryption KMS key to use to encrypt the volume. If you do not specify a KMS key, the KMS key that is used for encryption depends on the encryption state of the source snapshot and its ownership. For more information, see the encryption outcomes table.

> **Note**
>
> If you are using the API or AWS CLI to specify a KMS key, be aware that AWS authenticates the KMS key asynchronously. If you specify a KMS key ID, an alias, or an ARN that is not valid, the action can appear to complete, but it eventually fails.

You cannot change the KMS key that is associated with an existing snapshot or volume. However, you can associate a different KMS key during a snapshot copy operation so that the resulting copied snapshot is encrypted by the new KMS key.

Encrypt an empty volume on creation

When you create a new, empty EBS volume, you can encrypt it by enabling encryption for the specific volume creation operation. If you enabled EBS encryption by default, the volume is automatically encrypted using your default KMS key for EBS encryption. Alternatively, you can specify a different symmetric encryption KMS key for the specific volume creation operation. The volume is encrypted by the time it is first available, so your data is always secured. For detailed procedures, see Create an Amazon EBS volume.

By default, the KMS key that you selected when creating a volume encrypts the snapshots that you make from the volume and the volumes that you restore from those encrypted snapshots. You cannot remove encryption from an encrypted volume or snapshot, which means that a volume restored from an encrypted snapshot, or a copy of an encrypted snapshot, is always encrypted.

Public snapshots of encrypted volumes are not supported, but you can share an encrypted snapshot with specific accounts. For detailed directions, see Share an Amazon EBS snapshot with other AWS accounts.

Encrypt unencrypted resources

You can't directly encrypt existing unencrypted volumes or snapshots.

To encrypt an unencrypted volume, create a snapshot of that volume, and then use the snapshot to create a new encrypted volume. For more information, see Create snapshots and Create a volume.

To encrypt an unencrypted snapshot, create an encrypted copy of that snapshot. For more information, see Copy a snapshot.

If you enable your account for encryption by default, volumes and snapshot copies created from unencrypted snapshots are always encrypted. Otherwise, you must specify the encryption parameters in the request. For more information, see Enable encryption by default.

Rotate AWS KMS keys used for Amazon EBS encryption

Cryptographic best practices discourage extensive reuse of encryption keys.

To create new cryptographic material for use with Amazon EBS encryption, you can either create a new customer managed key, and then change your applications to use that new KMS key. Or, you can enable automatic key rotation for an existing customer managed key.

When you enable automatic key rotation for a customer managed key, AWS KMS generates new cryptographic material for the KMS key every year. AWS KMS saves all previous versions of the cryptographic material so that you can continue to decrypt and use volumes and snapshots previously encrypted with that KMS key material. AWS KMS does not delete any rotated key material until you delete the KMS key.

When you use a rotated customer managed key to encrypt a new volume or snapshot, AWS KMS uses the current (new) key material. When you use a rotated customer managed key to decrypt a volume or snapshot, AWS KMS uses the version of the cryptographic material that was used to encrypt it. If a volume or snapshot is encrypted with a previous version of the cryptographic material, AWS KMS continues to use that previous version to decrypt it. AWS KMS does not reencrypt previously encrypted volumes or snapshots to use the new cryptographic material after a key rotation. They remain encrypted with the cryptographic material with which they were originally encrypted. You can safely use a rotated customer managed key in applications and AWS services without code changes.

> **Note**
>
> - Automatic key rotation is supported only for symmetric customer managed keys with key material that AWS KMS creates.
> - AWS KMS automatically rotates AWS managed keys every year. You can't enable or disable key rotation for AWS managed keys.

For more information, see [Rotating KMS key] in the *AWS Key Management Service Developer Guide*.

Amazon EBS encryption examples

When you create an encrypted EBS resource, it is encrypted by your account's default KMS key for EBS encryption unless you specify a different customer managed key in the volume creation parameters or the block device mapping for the AMI or instance.

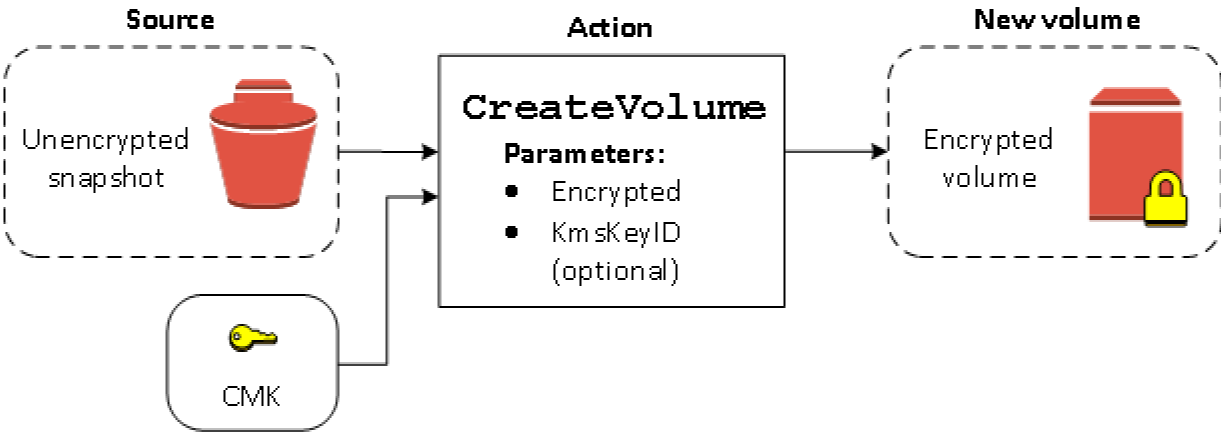
The following examples illustrate how you can manage the encryption state of your volumes and snapshots. For a full list of encryption cases, see the encryption outcomes table.

Examples

- Restore an unencrypted volume (encryption by default not enabled)
- Restore an unencrypted volume (encryption by default enabled)
- Copy an unencrypted snapshot (encryption by default not enabled)
- Copy an unencrypted snapshot (encryption by default enabled)
- Re-encrypt an encrypted volume
- Re-encrypt an encrypted snapshot
- Migrate data between encrypted and unencrypted volumes
- Encryption outcomes

Restore an unencrypted volume (encryption by default not enabled)

Without encryption by default enabled, a volume restored from an unencrypted snapshot is unencrypted by default. However, you can encrypt the resulting volume by setting the `Encrypted` parameter and, optionally, the `KmsKeyId` parameter. The following diagram illustrates the process.

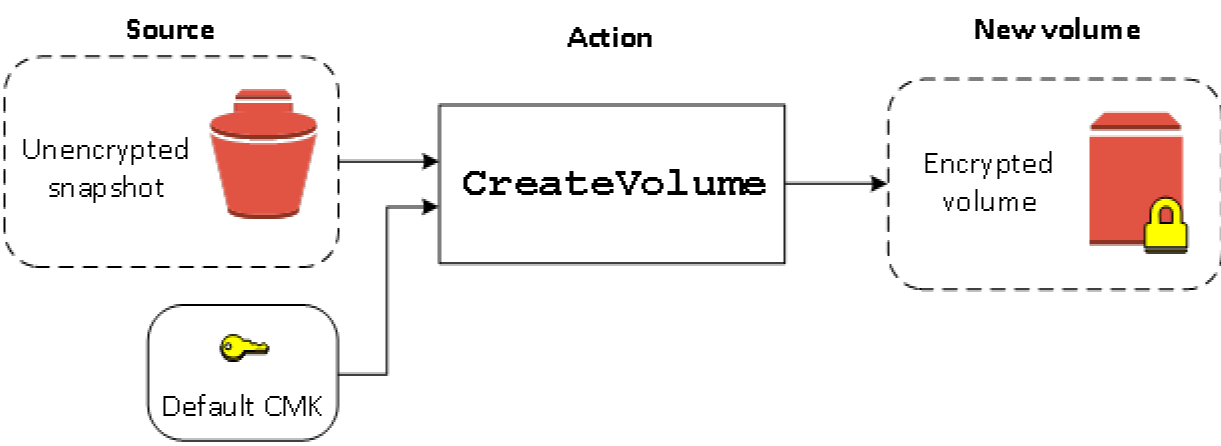


If you leave out the `KmsKeyId` parameter, the resulting volume is encrypted using your default KMS key for EBS encryption. You must specify a KMS key ID to encrypt the volume to a different KMS key.

For more information, see [Create an Amazon EBS volume](#).

Restore an unencrypted volume (encryption by default enabled)

When you have enabled encryption by default, encryption is mandatory for volumes restored from unencrypted snapshots, and no encryption parameters are required for your default KMS key to be used. The following diagram shows this simple default case:

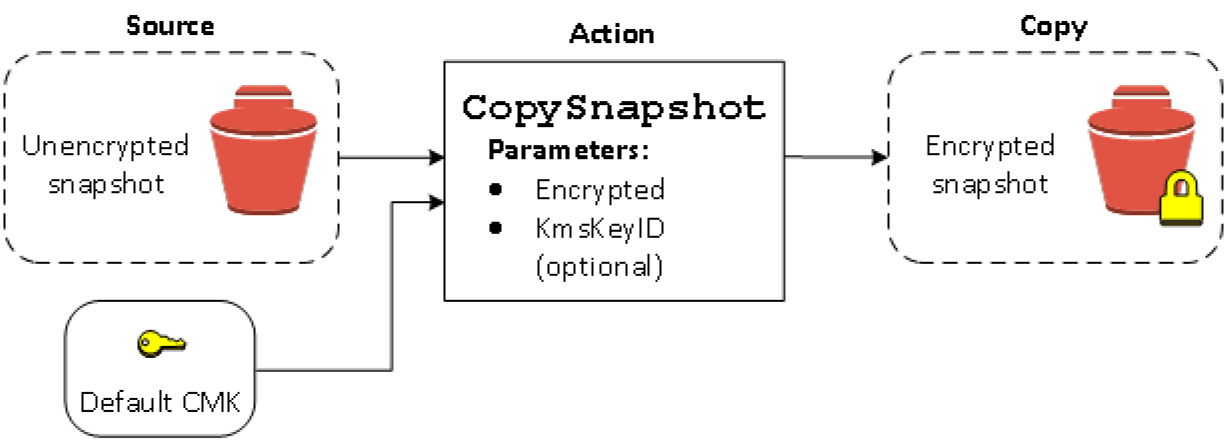


If you want to encrypt the restored volume to a symmetric customer managed encryption key, you must supply both the `Encrypted` and `KmsKeyId` parameters as shown in [Restore an unencrypted volume (encryption by default not enabled)](#).

Copy an unencrypted snapshot (encryption by default not enabled)

Without encryption by default enabled, a copy of an unencrypted snapshot is unencrypted by default. However, you can encrypt the resulting snapshot by setting the `Encrypted` parameter and, optionally, the `KmsKeyId` parameter. If you omit `KmsKeyId`, the resulting snapshot is encrypted by your default KMS key. You must specify a KMS key ID to encrypt the volume to a different symmetric encryption KMS key.

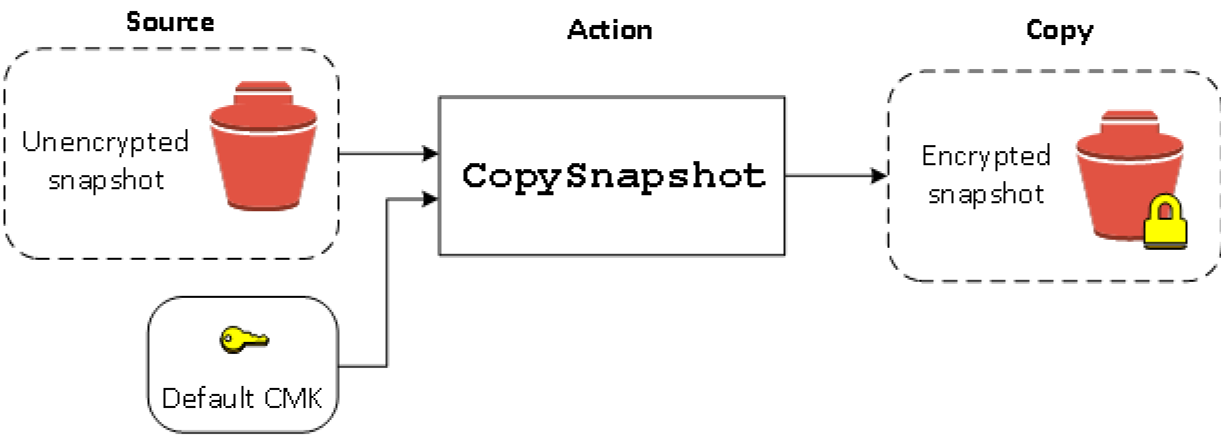
The following diagram illustrates the process.



You can encrypt an EBS volume by copying an unencrypted snapshot to an encrypted snapshot and then creating a volume from the encrypted snapshot. For more information, see [Copy an Amazon EBS snapshot](#).

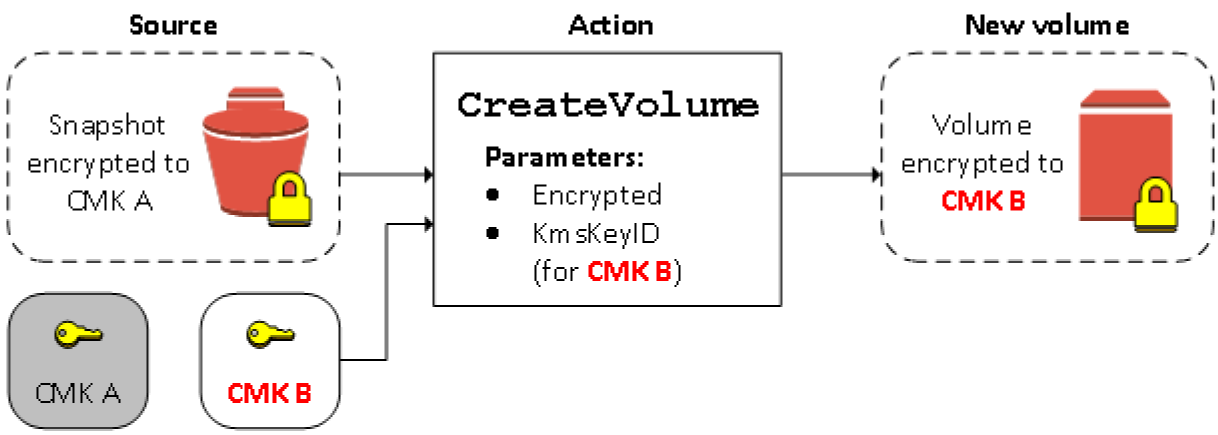
Copy an unencrypted snapshot (encryption by default enabled)

When you have enabled encryption by default, encryption is mandatory for copies of unencrypted snapshots, and no encryption parameters are required if your default KMS key is used. The following diagram illustrates this default case:



Re-encrypt an encrypted volume

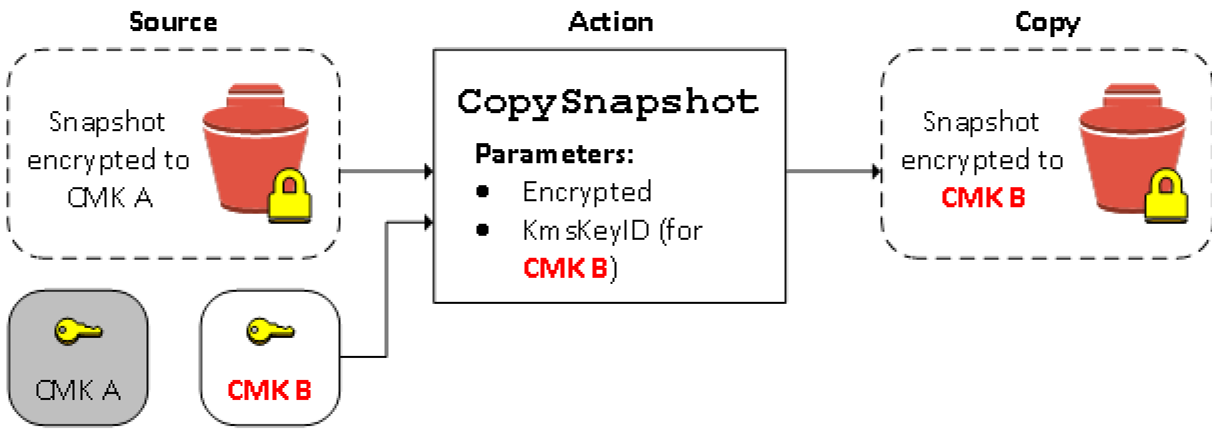
When the `CreateVolume` action operates on an encrypted snapshot, you have the option of re-encrypting it with a different KMS key. The following diagram illustrates the process. In this example, you own two KMS keys, KMS key A and KMS key B. The source snapshot is encrypted by KMS key A. During volume creation, with the KMS key ID of KMS key B specified as a parameter, the source data is automatically decrypted, then re-encrypted by KMS key B.



For more information, see Create an Amazon EBS volume.

Re-encrypt an encrypted snapshot

The ability to encrypt a snapshot during copying allows you to apply a new symmetric encryption KMS key to an already-encrypted snapshot that you own. Volumes restored from the resulting copy are only accessible using the new KMS key. The following diagram illustrates the process. In this example, you own two KMS keys, KMS key A and KMS key B. The source snapshot is encrypted by KMS key A. During copy, with the KMS key ID of KMS key B specified as a parameter, the source data is automatically re-encrypted by KMS key B.



In a related scenario, you can choose to apply new encryption parameters to a copy of a snapshot that has been shared with you. By default, the copy is encrypted with a KMS key shared by the snapshot's owner. However, we recommend that you create a copy of the shared snapshot using a different KMS key that you control. This protects your access to the volume if the original KMS key is compromised, or if the owner revokes the KMS key for any reason. For more information, see Encryption and snapshot copying.

Migrate data between encrypted and unencrypted volumes

When you have access to both an encrypted and unencrypted volume, you can freely transfer data between them. EC2 carries out the encryption and decryption operations transparently.

Linux instances

For example, use the **rsync** command to copy the data. In the following command, the source data is located in `/mnt/source` and the destination volume is mounted at `/mnt/destination`.

```
[ec2-user ~]$ sudo rsync -avh --progress /mnt/source/ /mnt/destination/
```

Windows instances

For example, use the **robocopy** command to copy the data. In the following command, the source data is located in `D:\` and the destination volume is mounted at `E:\`.

```
PS C:\> robocopy D:\sourcefolder E:\destinationfolder /e /copyall /eta
```

We recommend using folders rather than copying an entire volume, as this avoids potential problems with hidden folders.

Encryption outcomes

The following table describes the encryption outcome for each possible combination of settings.

Is encryption enabled?	Is encryption by default enabled?	Source of volume	Default (no customer managed key specified)	Custom (customer managed key specified)
No	No	New (empty) volume	Unencrypted	N/A
No	No	Unencrypted snapshot that you own	Unencrypted	
No	No	Encrypted snapshot that you own	Encrypted by same key	
No	No	Unencrypted snapshot that is shared with you	Unencrypted	
No	No	Encrypted snapshot that is shared with you	Encrypted by default customer managed key*	
Yes	No	New volume	Encrypted by default customer managed key	Encrypted by a specified customer managed key**
Yes	No	Unencrypted snapshot that you own	Encrypted by default customer managed key	
Yes	No	Encrypted snapshot that you own	Encrypted by same key	
Yes	No	Unencrypted snapshot that is shared with you	Encrypted by default customer managed key	

Is encryption enabled?	Is encryption by default enabled?	Source of volume	Default (no customer managed key specified)	Custom (customer managed key specified)
Yes	No	Encrypted snapshot that is shared with you	Encrypted by default customer managed key	
No	Yes	New (empty) volume	Encrypted by default customer managed key	N/A
No	Yes	Unencrypted snapshot that you own	Encrypted by default customer managed key	
No	Yes	Encrypted snapshot that you own	Encrypted by same key	
No	Yes	Unencrypted snapshot that is shared with you	Encrypted by default customer managed key	
No	Yes	Encrypted snapshot that is shared with you	Encrypted by default customer managed key	
Yes	Yes	New volume	Encrypted by default customer managed key	Encrypted by a specified customer managed key
Yes	Yes	Unencrypted snapshot that you own	Encrypted by default customer managed key	
Yes	Yes	Encrypted snapshot that you own	Encrypted by same key	

Is encryption enabled?	Is encryption by default enabled?	Source of volume	Default (no customer managed key specified)	Custom (customer managed key specified)
Yes	Yes	Unencrypted snapshot that is shared with you	Encrypted by default customer managed key	
Yes	Yes	Encrypted snapshot that is shared with you	Encrypted by default customer managed key	

* This is the default customer managed key used for EBS encryption for the AWS account and Region. By default this is a unique AWS managed key for EBS, or you can specify a customer managed key.

** This is a customer managed key specified for the volume at launch time. This customer managed key is used instead of the default customer managed key for the AWS account and Region.

Amazon EBS volume performance

Several factors, including I/O characteristics and the configuration of your instances and volumes, can affect the performance of Amazon EBS. If you follow the guidance on our Amazon EBS and Amazon EC2 product detail pages you'll usually achieve good performance. However, there are some cases where you might need to do some tuning to achieve peak performance. We recommend that you tune performance with information from your actual workload, in addition to benchmarking, to determine your optimal configuration. After you learn the basics of working with EBS volumes, it's a good idea to look at the I/O performance you require and at your options for increasing Amazon EBS performance to meet those requirements.

AWS updates to the performance of EBS volume types might not immediately take effect on your existing volumes. To see full performance on an older volume, you might first need to perform a `ModifyVolume` action on it. For more information, see Modify an Amazon EBS volume using Elastic Volumes operations.

Contents

Amazon EBS performance tips

These tips represent best practices for getting optimal performance from your EBS volumes in a variety of user scenarios.

Use EBS-optimized instances

On instances without support for EBS-optimized throughput, network traffic can contend with traffic between your instance and your EBS volumes; on EBS-optimized instances, the two types of traffic are kept separate. Some EBS-optimized instance configurations incur an extra cost (such as C3, R3, and M3), while others are always EBS-optimized at no extra cost (such as M4, C4, C5, and D2). For more information, see Amazon EBS optimization.

Configure instance bandwidth

For supported instance types, you can configure the instance bandwidth weighting to increase Amazon EBS bandwidth by 25 percent using the `ebs-1` bandwidth weighting. This feature allows you to optimize your instance's network resource allocation between EBS and VPC networking, potentially improving EBS performance for I/O-intensive workloads. For more information, see Configurable instance bandwidth weighting.

Understand how performance is calculated

When you measure the performance of your EBS volumes, it is important to understand the units of measure involved and how performance is calculated. For more information, see Amazon EBS I/O characteristics and monitoring.

Understand your workload

There is a relationship between the maximum performance of your EBS volumes, the size and number of I/O operations, and the time it takes for each action to complete. Each of these factors (performance, I/O, and latency) affects the others, and different applications are more sensitive to one factor or another. For more information, see Benchmark Amazon EBS volumes.

Be aware of the performance penalty When initializing volumes from snapshots

There is a significant increase in latency when you first access each block of data on a new EBS volume that was created from a snapshot. You can avoid this performance hit using one of the following options:

- Access each block prior to putting the volume into production. This process is called *initialization* (formerly known as pre-warming). For more information, see Manually initialize the volumes after creation.
- Enable fast snapshot restore on a snapshot to ensure that the EBS volumes created from it are fully-initialized at creation and instantly deliver all of their provisioned performance. For more information, see Amazon EBS fast snapshot restore.

Factors that can degrade HDD performance

When you create a snapshot of a Throughput Optimized HDD (`st1`) or Cold HDD (`sc1`) volume, performance may drop as far as the volume's baseline value while the snapshot is in progress. This behavior is specific to these volume types. Other factors that can limit performance include driving more throughput than the instance can support, the performance penalty encountered while initializing volumes created from a snapshot, and excessive amounts of small, random I/O on the volume. For more information about calculating throughput for HDD volumes, see Amazon EBS volume types.

Your performance can also be impacted if your application isn't sending enough I/O requests. This can be monitored by looking at your volume's queue length and I/O size. The queue length is the number of pending I/O requests from your application to your volume. For maximum consistency, HDD-backed volumes must maintain a queue length (rounded to the nearest whole number) of 4 or more when performing 1 MiB sequential I/O. For more information about ensuring consistent performance of your volumes, see Amazon EBS I/O characteristics and monitoring

Increase read-ahead for high-throughput, read-heavy workloads on `st1` and `sc1` (*Linux instances only*)

Some workloads are read-heavy and access the block device through the operating system page cache (for example, from a file system). In this case, to achieve the maximum throughput, we recommend that you configure the read-ahead setting to 1 MiB. This is a per-block-device setting that should only be applied to your HDD volumes.

To examine the current value of read-ahead for your block devices, use the following command:

```
$ sudo blockdev --report /dev/<device>
```

Block device information is returned in the following format:

```
RO    RA   SSZ   BSZ   StartSec            Size   Device
rw   256   512  4096       4096      8587820544   /dev/<device>
```

The device shown reports a read-ahead value of 256 (the default). Multiply this number by the sector size (512 bytes) to obtain the size of the read-ahead buffer, which in this case is 128 KiB. To set the buffer value to 1 MiB, use the following command:

```
$ sudo blockdev --setra 2048 /dev/<device>
```

Verify that the read-ahead setting now displays 2,048 by running the first command again.

Only use this setting when your workload consists of large, sequential I/Os. If it consists mostly of small, random I/Os, this setting will actually degrade your performance. In general, if your workload consists mostly of small or random I/Os, you should consider using a General Purpose SSD (gp2 and gp3) volume rather than an st1 or sc1 volume.

Use a modern Linux kernel (*Linux instances only*)

Use a modern Linux kernel with support for indirect descriptors. Any Linux kernel 3.8 and above has this support, as well as any current-generation EC2 instance. If your average I/O size is at or near 44 KiB, you may be using an instance or kernel without support for indirect descriptors. For information about deriving the average I/O size from Amazon CloudWatch metrics, see [Amazon EBS I/O characteristics and monitoring].

To achieve maximum throughput on st1 or sc1 volumes, we recommend applying a value of 256 to the xen_blkfront.max parameter (for Linux kernel versions below 4.6) or the xen_blkfront.max_indirect_segments parameter (for Linux kernel version 4.6 and above). The appropriate parameter can be set in your OS boot command line.

For example, in an Amazon Linux AMI with an earlier kernel, you can add it to the end of the kernel line in the GRUB configuration found in /boot/grub/menu.lst:

```
kernel /boot/vmlinuz-4.4.5-15.26.amzn1.x86_64 root=LABEL=/ console=ttyS0
 xen_blkfront.max=256
```

For a later kernel, the command would be similar to the following:

```
kernel /boot/vmlinuz-4.9.20-11.31.amzn1.x86_64 root=LABEL=/ console=tty1 console=ttyS0
 xen_blkfront.max_indirect_segments=256
```

Reboot your instance for this setting to take effect.

For more information, see [Configure GRUB for paravirtual AMIs]. Other Linux distributions, especially those that do not use the GRUB boot loader, may require a different approach to adjusting the kernel parameters.

For more information about EBS I/O characteristics, see the Amazon EBS: Designing for Performance re:Invent presentation on this topic.

Use RAID 0 to maximize utilization of instance resources

Some instance types can drive more I/O throughput than what you can provision for a single EBS volume. You can join multiple volumes together in a RAID 0 configuration to use the available bandwidth for these instances. For more information, see Amazon EBS and RAID configuration.

Monitor Amazon EBS volume performance

You can monitor and analyze the performance of your Amazon EBS volumes using Amazon CloudWatch, status checks, and EBS detailed performance statistics. For more information, see Amazon CloudWatch metrics for Amazon EBS and Amazon EBS detailed performance statistics.

Amazon EBS optimization

An Amazon EBS–optimized instance uses an optimized configuration stack and provides additional, dedicated capacity for Amazon EBS I/O. This optimization provides the best performance for your EBS volumes by minimizing contention between Amazon EBS I/O and other traffic from your instance.

EBS–optimized instances deliver dedicated bandwidth to Amazon EBS. When attached to an EBS–optimized instance:

- `io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations. `io2` Block Express volumes also deliver better outlier latency compared to General Purpose volumes, reducing the frequency of I/Os exceeding 800 microseconds by over 10 times. `io1` and `io2` volumes are designed to deliver at least 90% of their provisioned IOPS performance 99.9% of the time in a given year.
- `gp2` and `gp3` volumes are designed to deliver at least 90% of their provisioned IOPS performance 99% of the time in a given year.
- `st1` and `sc1` volumes deliver at least 90% of their expected throughput performance 99% of the time in a given year.

Non-compliant periods are approximately uniformly distributed, targeting 99% of expected total throughput each hour. For more information, see Amazon EBS volume types.

The I/O size of your workload will impact the observed average latency as latency increases with larger I/O size. For example, `io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations.

For more information, see Amazon EBS–optimized instances in the *Amazon EC2 User Guide*.

Initialize Amazon EBS volumes

When you create an Amazon EBS volume, either from an EBS snapshot or from another EBS volume (volume copy), the data blocks must be written to the volume before you can access them. For volumes created from snapshots, the data blocks must be downloaded from Amazon S3 to the new volume. For volume copies, the data blocks must be copied from the source volume to the volume copy. This process is called *volume initialization*. During this time, the volume being initialized might experience increased I/O latency and decreased performance. Full volume performance is achieved only once all storage blocks have been downloaded and written to the volume.

Note

Empty volumes deliver their maximum performance immediately after creation and do not require initialization.

The default volume initialization rate fluctuates throughout the initialization process, which could make completion times unpredictable. To minimize the performance impacts associated with volume initialization, you could use the following options:

Note

volume initialization rate and fast snapshot restore are not supported for volume copies. For more information, see Volume copy initialization.

Topics

- Monitor the status of Amazon EBS volume initialization

Use an Amazon EBS Provisioned Rate for Volume Initialization

When you create an Amazon EBS volume from a snapshot, you can optionally specify an Amazon EBS Provisioned Rate for Volume Initialization (volume initialization rate) that ranges from 100 to 300 MiB/s. If you specify a volume initialization rate, the snapshot blocks are downloaded from Amazon S3 and written to the volume at the specified rate after creation. This enables you to create volumes that become fully initialized and fully performant in a predictable amount of time.

Using a volume initialization rate is especially useful when you are creating multiple volumes simultaneously and you need all of them to be initialized in a predictable amount of time.

> **Note**
>
> Amazon EBS Provisioned Rate for Volume Initialization is supported with all Amazon EBS volume types, and all Amazon EC2 instance types, including Amazon EC2 Mac instances.

You can specify a volume initialization rate:

- For individual volume creation requests
- For EBS volume block device mappings in instance launch requests
- For EBS volume block device mappings in launch templates
- For EBS volumes created by root volume replacement tasks
- For EBS volumes on Amazon EKS clusters (created by EBS CSI Driver) and Amazon ECS clusters

Topics

How it works

When you create a volume with a volume initialization rate, the snapshot blocks are downloaded from Amazon S3 to the volume at the rate you specify.

The amount of time taken to initialize the volume depends on the following:

- The size of the snapshot data, not the size of the volume being created.

> **Tip**
>
> To find a snapshot's data size, check the `FullSnapshotSizeInBytes` field in the describe-snapshots command output, or the **Full snapshot size** field in the console.

- The volume initialization rate that you specify

For example, if you create a 20 GiB volume using a snapshot that has 10 GiB of data, and you specify a volume initialization rate of 300 MiB/s, the volume will be fully initialized in approximately 34.1 seconds (10 GiB / 300 MiB/s = 34.1 seconds). Similarly, if you create 10 volumes with that same snapshot and volume initialization rate concurrently, all 10 volumes will be fully initialized in 34.1 seconds.

Considerations

- You can specify a volume initialization rate of between 100 and 300 MiB/s.
- When you specify a volume initialization rate, the charges and completion time are based on the size of the snapshot data (not the size of the volume) and the rate you specify. For more information, see Billing.
- Amazon EBS delivers an average rate that is within 10 percent of the volume initialization rate that you specify for 99 percent of the time.
- If you specify a volume initialization rate and use a snapshot that is enabled for fast snapshot restore, Amazon EBS uses the specified rate instead of fast snapshot restore. To use fast snapshot restore instead, do not specify a volume initialization rate.
- If Amazon EBS can't initialize the volume at the specified volume initialization rate due to capacity constraints or because you have exceeded your quota, the request fails.
- You can't specify a volume initialization rate for volumes created on AWS Outposts, or in Local Zones or Wavelength Zones.

Quotas

There is a limit of 5,000 MiB/s on the cumulative volume initialization rate that you can request across concurrent volume creation requests. For example, you can make 50 concurrent volume creation requests with a rate of 100 MiB/s (50 simultaneous requests * 100 MiB/s rate), or 25 concurrent requests with a rate of 200 MiB/s (25 simultaneous requests * 200 MiB/s rate). This limit applies on a per Region basis. If a request exceeds this limit, it fails. Either wait for some of the in-progress requests to complete or request a quota increase. For more information, see Quotas for Amazon EBS.

Billing

When you create a volume with a volume initialization rate, you are charged a rate per GiB of snapshot data, per MiB of specified initialization rate. The rate varies by Region. For more information, see Amazon EBS pricing.

You are charged based on the size of the snapshot data, not the size of the volume. For example, if you create a snapshot of a volume that is 100 GiB in size, but has only 50 GiB of data, the snapshot has a volume size of 100 GiB, but the snapshot data size is 50 GiB. If you use that snapshot to create a volume and specify a volume initialization rate, your charges are based on the 50 GiB of snapshot data.

> **Tip**
>
> To find a snapshot's data size, check the `FullSnapshotSizeInBytes` field in the describe-snapshots command output, or the **Full snapshot size** field in the console.

The formula is as follows:

```
rate for Region x snapshot data size x volume initialization rate
```

You are billed the full amount as soon as the volume enters the `active` state. Failed requests are not billed.

If you delete a volume before the volume initialization completes, you are still billed for the requested volume initialization rate.

Use a snapshot that is enabled for fast snapshot restore

If you create a volume from a snapshot that is enabled for fast snapshot restore, the volume is fully initialized at creation and it immediately delivers its full performance. For more information about using fast snapshot restore, see Amazon EBS fast snapshot restore.

Manually initialize the volumes after creation

You can manually initialize an Amazon EBS volume after creation to help minimize the performance impacts of volume initialization.

You can use the following procedures to manually initialize an Amazon EBS volume after creation.

> **Important**
>
> While initializing Provisioned IOPS SSD volumes that were created from snapshots, the performance of the volume may drop below 50 percent of its expected level, which causes the volume to display a `warning` state in the **I/O Performance** status check. This is expected, and you can ignore the `warning` state on Provisioned IOPS SSD volumes while you are initializing them. For more information, see Amazon EBS volume status checks.

Linux instances

To initialize a volume created from a snapshot on Linux

1. Attach the newly-restored volume to your Linux instance.
2. Use the **lsblk** command to list the block devices on your instance.

   ```
   $ lsblk
   NAME  MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
   xvdf  202:80   0  30G  0 disk
   xvda1 202:1    0   8G  0 disk /
   ```

   Here you can see that the new volume, `/dev/xvdf`, is attached, but not mounted (because there is no path listed under the `MOUNTPOINT` column).
3. Use the **dd** or **fio** utilities to read all of the blocks on the device. The **dd** command is installed by default on Linux systems, but **fio** is considerably faster because it allows multi-threaded reads.

> **Note**
>
> This step may take several minutes up to several hours, depending on your EC2 instance bandwidth, the IOPS provisioned for the volume, and the size of the volume.

[**dd**] The `if` (input file) parameter should be set to the drive you wish to initialize. The `of` (output file) parameter should be set to the Linux null virtual device, `/dev/null`. The `bs` parameter sets the block size of the read operation; for optimal performance, this should be set to 1 MB.

> **Important**
>
> Incorrect use of **dd** can easily destroy a volume's data. Be sure to follow precisely the example command below. Only the `if=/dev/xvdf` parameter will vary depending on the name of the device you are reading.

```
$ sudo dd if=/dev/xvdf of=/dev/null bs=1M
```

[**fio**] If you have **fio** installed on your system, use the following command to initialize your volume. The `--filename` (input file) parameter should be set to the drive you wish to initialize.

```
$ sudo fio --filename=/dev/xvdf --rw=read --bs=1M --iodepth=32 --ioengine=libaio --direct=1 --name=volume-initialize
```

To install **fio** on Amazon Linux, use the following command:

```
sudo yum install -y fio
```

To install **fio** on Ubuntu, use the following command:

```
sudo apt-get install -y fio
```

When the operation is finished, you will see a report of the read operation. Your volume is now ready for use. For more information, see Make an Amazon EBS volume available for use.

Windows instances

Before using either tool, gather information about the disks on your system as follows:

To gather information about the system disks

1. Use the **wmic** command to list the available disks on your system:

```
wmic diskdrive get size,deviceid
```

The following is example output:

```
DeviceID            Size
\\.\PHYSICALDRIVE2  80517265920
\\.\PHYSICALDRIVE1  80517265920
\\.\PHYSICALDRIVE0  128849011200
\\.\PHYSICALDRIVE3  107372805120
```

2. Identify the disk to initialize using **dd** or **fio**. The `C:` drive is on `\\.\PHYSICALDRIVE0`. You can use the `diskmgmt.msc` utility to compare drive letters to disk drive numbers if you are not sure which drive number to use.

Use the dd utility

Complete the following procedures to install and use **dd** to initialize a volume.

Important considerations

- Initializing a volume takes from several minutes up to several hours, depending on your EC2 instance bandwidth, the IOPS provisioned for the volume, and the size of the volume.
- Incorrect use of **dd** can easily destroy a volume's data. Be sure to follow this procedure precisely.

To install dd for Windows

The **dd** for Windows program provides a similar experience to the **dd** program that is commonly available for Linux and Unix systems, and it enables you to initialize Amazon EBS volumes that have been created from snapshots. The most recent beta versions support the `/dev/null` virtual device. If you install an earlier version, you can use the `nul` virtual device instead. Full documentation is available at http://www.chrysocome.net/dd.

1. Download the most recent binary version of **dd** for Windows from http://www.chrysocome.net/dd.
2. (Optional) Create a folder for command line utilities that is easy to locate and remember, such as `C:\bin`. If you already have a designated folder for command line utilities, you can use that folder instead in the following step.
3. Unzip the binary package and copy the `dd.exe` file to your command line utilities folder (for example, `C:\bin`).
4. Add the command line utilities folder to your Path environment variable so you can run the programs in that folder from anywhere.
   a. Choose **Start**, open the context (right-click) menu for **Computer**, and then choose **Properties**.
   b. Choose **Advanced system settings**, **Environment Variables**.
   c. For **System Variables**, select the variable **Path** and choose **Edit**.
   d. For **Variable value**, append a semicolon and the location of your command line utility folder (**`;C:\bin\`**) to the end of the existing value.
   e. Choose **OK** to close the **Edit System Variable** window.
5. Open a new command prompt window. The previous step doesn't update the environment variables in your current command prompt windows. The command prompt windows that you open now that you completed the previous step are updated.

To initialize a volume using dd for Windows

Run the following command to read all blocks on the specified device (and send the output to the `/dev/null` virtual device). This command safely initializes your existing data.

```
dd if=\\.\PHYSICALDRIVEn of=/dev/null bs=1M --progress --size
```

You might get an error if **dd** attempts to read beyond the end of the volume. You can safely ignore this error.

If you used an earlier version of the **dd** command, it does not support the /dev/null device. Instead, you can use the nul device as follows.

```
dd if=\\.\PHYSICALDRIVEn of=nul bs=1M --progress --size
```

Use the fio utility

Complete the following procedures to install and use **fio** to initialize a volume.

To install fio for Windows

The **fio** for Windows program provides a similar experience to the **fio** program that is commonly available for Linux and Unix systems, and it allows you to initialize Amazon EBS volumes created from snapshots. For more information, see https://github.com/axboe/fio.

1. Download the **fio** MSI installer by expanding **Assets** for the latest release and selecting the MSI installer.
2. Install **fio**.

To initialize a volume using fio for Windows

1. Run a command similar to the following to initialize a volume:

```
fio --filename=\\.\PHYSICALDRIVEn  --rw=read --bs=1M --iodepth=32 --direct=1 --name=volume-initialize
```

2. When the operation completes, you are ready to use your new volume. For more information, see Make an Amazon EBS volume available for use.

Monitor the status of Amazon EBS volume initialization

When you create a volume, either from a snapshot or from another volume (volume copy), you can monitor the status of the volume initialization to determine whether the initialization process is complete. You can monitor volume initialization using the following options:

Topics

- AWS CLI and Amazon EC2 console
- Amazon EventBridge

AWS CLI and Amazon EC2 console

You can use the AWS CLI and Amazon EC2 console to check the status of the volume initialization at any time after the volume has been created. The following information is provided:

- **Initialization type** (AWS CLI only) — Indicates the type of volume initialization used. `default` for fast snapshot restore and default volume initialization, `provisioned-rate` for Amazon EBS Provisioned Rate for Volume Initialization, and `volume-copy` for volume copy initialization.
- **Estimated time to completion** (AWS CLI only) — Only for volumes created using a Amazon EBS Provisioned Rate for Volume Initialization. The estimated remaining time, in seconds, for the volume initialization to complete.
- **Progress** — The progress, as a percentage (0-100), for the volume initialization process. For volumes initialized with fast snapshot restore, the progress moves to 100 percent immediately after creation.
- **Initialization state** — The overall state of the volume initialization (`initializing` or `completed`). For volumes initialized with fast snapshot restore, the state moves to `completed` immediately after creation.

> **Note**
>
> It can take up to 5 minutes for the volume initialization information to be updated.

Console

To monitor status of volume initialization

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Volumes**.
3. Select the volume for which to check the volume initialization status.
4. The **Initialization state** field in the grid and **Details** tab provide progress information in the following format: *Initialization state (progress percentage)*. For example, *Initializing (75%)*.

The possible initialization states include: *Initializing* and *Completed*.

AWS CLI

To monitor status of volume initialization

Use the [describe-volume-status](https://docs.aws.amazon.com/cli/latest/reference/ec2/describe-volume-status.html) AWS CLI command to view the initialization status. `EstimatedTimeToCompleteInSeconds` is returned only for volumes created with an Amazon EBS Provisioned Rate for Volume Initialization.

For example, the following command checks the initialization status for volume `vol-11111111111111111`, which was created with an Amazon EBS Provisioned Rate for Volume Initialization.

```
aws ec2 describe-volume-status --volume-id vol-01111111111111111
```

The following is example output.

```
{
    "VolumeStatuses": [
        {
            "Actions": [],
            "AvailabilityZone": "us-east-1a",
            "Events": [],
            "VolumeID": "vol-11111111111111111",
            "VolumeStatus": {
                "Details": [
                    {
                        "Name": "io-enabled",
                        "Status": "passed"
                    },
                    {
                        "Name": "io-performance",
                        "Status": "not-applicable"
                    },
                    {
                        "Name": "initialization-state",
                        "Status": "completed"
                    }
                ],
                "Status": "ok"
```

```
            },
            "InitializationStatusDetails": {
                "InitializationType": "provisioned-rate",
                "Progress": 75,
                "EstimatedTimeToCompleteInSeconds": 850
            }
        }
    ]
}
```

Amazon EventBridge

An Amazon EventBridge event is sent to your account within five minutes **after** the volume initialization has completed. You can create rules that trigger programmatic actions in response to these events.

> **Note**
>
> - Events are emitted on a best effort basis.
> - If you delete the volume before initialization completes, or within 5 minutes after initialization completes, you might not receive the event.

For more information about the event, see [EBS volume initialization event](#).

To monitor status of volume initialization using EventBridge

1. Open the Amazon EventBridge console at https://console.aws.amazon.com/events/.
2. Choose **Rules**, **Create rule**.
3. For **Step 1**, do the following:
   a. Specify a name and description for the rule.
   b. For **Event bus**, choose the bus to receive the events. If you haven't created a custom event bus, keep **default**, or see Creating an event bus.
   c. For **Rule type**, keep **Rule with an event pattern**.
   d. Choose **Next**.
4. For **Step 2**, do the following:

a. For **Event source**, keep **AWS events or EventBridge partner events**.

b. For **Creation method**, choose **Custom pattern (JSON editor)**.

c. For **Event pattern**, add the following:

```
{
    "detail-type": ["EBS Volume Notification"],
    "source": ["aws.ec2"],
    "detail": {
        "event": ["initializeVolume"],
        "result": ["succeeded"]
    }
}
```

For an example event, see EBS volume initialization event.

d. Choose **Next**.

5. For **Step 3**, do the following:

a. For **Target types**, choose **AWS service**.

b. For **Select target**, choose **SNS topic**, and for **Topic** select the required topic. If you haven't created any topics, see Creating a topic.

c. For **Permissions**, keep **Use execution role (recommended)** selected.

d. For **Execution role**, keep **Create a new role for this specific resource** selected and the defauly role name.

e. Choose **Next**.

6. For **Step 4**, specify tags for the rule if needed, and then choose **Next**.

7. For **Step 5**, review the rule and then choose **Create rule**.

Configurable instance bandwidth weighting

Instance bandwidth configuration (IBC) is a feature that enables you to adjust the allocation of network bandwidth between Amazon EBS and VPC networking for an Amazon EC2 instance. This feature can help you optimize performance for workloads with specific bandwidth requirements. Instance bandwidth configuration is supported on some instances only. For more information, see Instance bandwidth weighting configuration.

For EBS performance, using the `ebs-1` bandwidth weighting increases the baseline EBS bandwidth by 25 percent while reducing the VPC networking bandwidth by the same absolute amount. This can be beneficial for I/O-intensive workloads that require higher EBS throughput.

When planning your workload, carefully consider your I/O size and patterns. Smaller I/O sizes are generally less affected by bandwidth limitations, while larger I/O sizes or sequential workloads can experience more significant impacts from bandwidth changes. It's crucial to thoroughly test your specific workload to ensure optimal performance with your chosen bandwidth weighting.

Considerations

- Configurable instance bandwidth is supported on select instance types. For more information, see Supported instance types.
- Using the `ebs-1` bandwidth weighting increases EBS bandwidth up to 25 percent, which can improve the performance of I/O-intensive applications. However, keep in mind that VPC networking bandwidth will be reduced by the same absolute amount (the combined bandwidth specification between EBS and networking does not change).
- Changes in bandwidth weighting can significantly affect I/O performance. With the `vpc-1` bandwidth weighting, network bandwidth is increased, but you might experience lower than expected IOPS for EBS volumes. This is because you might reach the EBS bandwidth limit before the IOPS limit, especially with larger I/O sizes. For example, an instance type that typically supports 240,000 IOPS with 16 KiB I/O size might achieve fewer IOPS when using `vpc-1` bandwidth weight due to the decreased EBS bandwidth.
- Always test your specific workload to ensure that your chosen bandwidth weighting meets your performance needs.
- You can configure the bandwidth weighting during instance launch or modify it for stopped instances. For more information see Configure bandwidth weighting for your instance.
- You can configure instance bandwidth weighting at no additional costs.

Amazon EBS I/O characteristics and monitoring

On a given volume configuration, certain I/O characteristics drive the performance behavior for your EBS volumes.

- SSD-backed volumes, General Purpose SSD (`gp2` and `gp3`) and Provisioned IOPS SSD (`io1` and `io2`), deliver consistent performance whether an I/O operation is random or sequential.

- HDD-backed volumes, Throughput Optimized HDD (`st1`) and Cold HDD (`sc1`), deliver optimal performance only when I/O operations are large and sequential.

To understand how SSD and HDD volumes will perform in your application, it is important to know the connection between demand on the volume, the quantity of IOPS available to it, the time it takes for an I/O operation to complete, and the volume's throughput limits.

Topics

IOPS

IOPS are a unit of measure representing input/output operations per second. The operations are measured in KiB, and the underlying drive technology determines the maximum amount of data that a volume type counts as a single I/O. I/O size is capped at 256 KiB for SSD volumes and 1,024 KiB for HDD volumes because SSD volumes handle small or random I/O much more efficiently than HDD volumes.

When small I/O operations are physically sequential, Amazon EBS attempts to merge them into a single I/O operation up to the maximum I/O size. Similarly, when I/O operations are larger than the maximum I/O size, Amazon EBS attempts to split them into smaller I/O operations. The following table shows some examples.

Volume type	Maximum I/O size	I/O operations from your application	Number of IOPS	Notes
SSD	256 KiB	1 x 1024 KiB I/O operation	4 (1,024÷256=4)	Amazon EBS splits the 1,024 KiB I/O operation into four

Volume type	Maximum I/O size	I/O operation s from your application	Number of IOPS	Notes
				smaller 256 KiB operations.
		8 x sequential 32 KiB I/O operations	1 (8x32=256)	Amazon EBS merges the eight sequential 32 KiB I/O operations into a single 256 KiB operation.
		8 random 32 KiB I/O operations	8	Amazon EBS counts random I/O operations separately.
HDD	1,024 KiB	1 x 1024 KiB I/O operation	1	The I/O operation is already equal to the maximum I/O size. It is not merged or split.
		8 x sequential 128 KiB I/O operations	1 (8x128=1,024)	Amazon EBS merges the eight sequential 128 KiB I/O operations into a single 1,024 KiB I/O operation.
		8 random 32 KiB I/O operations	8	Amazon EBS counts random I/O operations separately.

Consequently, when you create an SSD-backed volume supporting 3,000 IOPS (either by provisioning an `io1` or `io2` volume with 3,000 IOPS, by sizing a `gp2` volume at 1,000 GiB, or by using a `gp3` volume), and you attach it to an EBS-optimized instance that can provide sufficient bandwidth, you can transfer up to 3,000 I/Os of data per second, with throughput determined by I/O size.

Volume queue length and latency

The volume queue length is the number of pending I/O requests for a device. Latency is the true end-to-end client time of an I/O operation, in other words, the time elapsed between sending an I/O to EBS and receiving an acknowledgement from EBS that the I/O read or write is complete. Queue length must be correctly calibrated with I/O size and latency to avoid creating bottlenecks either on the guest operating system or on the network link to EBS.

Optimal queue length varies for each workload, depending on your particular application's sensitivity to IOPS and latency. If your workload is not delivering enough I/O requests to fully use the performance available to your EBS volume, then your volume might not deliver the IOPS or throughput that you have provisioned.

Transaction-intensive applications are sensitive to increased I/O latency and are well-suited for SSD-backed volumes. You can maintain high IOPS while keeping latency down by maintaining a low queue length and a high number of IOPS available to the volume. Consistently driving more IOPS to a volume than it has available can cause increased I/O latency. For maximum consistency, a volume must maintain an average queue depth (rounded to the nearest whole number) of one for every 1,000 provisioned IOPS in a minute. For example, for a volume provisioned with 3,000 IOPS, the queue depth average must be 3.

Throughput-intensive applications are less sensitive to increased I/O latency, and are well-suited for HDD-backed volumes. You can maintain high throughput to HDD-backed volumes by maintaining a high queue length when performing large, sequential I/O.

I/O size and volume throughput limits

For SSD-backed volumes, if your I/O size is very large, you may experience a smaller number of IOPS than you provisioned because you are hitting the throughput limit of the volume. For example, a `gp2` volume under 1,000 GiB with burst credits available has an IOPS limit of 3,000 and a volume throughput limit of 250 MiB/s. If you are using a 256 KiB I/O size, your volume reaches its throughput limit at 1000 IOPS (1000 x 256 KiB = 250 MiB). For smaller I/O sizes (such as 16 KiB), this same volume can sustain 3,000 IOPS because the throughput is well below 250 MiB/s. (These examples assume that your volume's I/O is not hitting the throughput limits of the instance.) For more information about the throughput limits for each EBS volume type, see Amazon EBS volume types.

For smaller I/O operations, you may see a higher-than-provisioned IOPS value as measured from inside your instance. This happens when the instance operating system merges small I/O operations into a larger operation before passing them to Amazon EBS.

If your workload uses sequential I/Os on HDD-backed `st1` and `sc1` volumes, you may experience a higher than expected number of IOPS as measured from inside your instance. This happens when the instance operating system merges sequential I/Os and counts them in 1,024 KiB-sized units. If your workload uses small or random I/Os, you may experience a lower throughput than you expect. This is because we count each random, non-sequential I/O toward the total IOPS count, which can cause you to hit the volume's IOPS limit sooner than expected.

Whatever your EBS volume type, if you are not experiencing the IOPS or throughput you expect in your configuration, ensure that your EC2 instance bandwidth is not the limiting factor. You should always use a current-generation, EBS-optimized instance (or one that includes 10 Gb/s network connectivity) for optimal performance. Another possible cause for not experiencing the expected IOPS is that you are not driving enough I/O to the EBS volumes.

Monitor I/O characteristics using CloudWatch

You can monitor these I/O characteristics with each volume's CloudWatch volume metrics.

Monitor for stalled I/O

`VolumeStalledIOCheck` monitors the status of your EBS volumes to determine when your volumes are impaired. The metric is a binary value that will return a `0` (pass) or a `1` (fail) status based on whether or not the EBS volume can complete I/O operations.

If the `VolumeStalledIOCheck` metric fails, you can either wait for AWS to resolve the issue, or you can take actions, such as replacing the affected volume or stopping and restarting the instance to which the volume is attached. In most cases, when this metric fails, EBS will automatically diagnose and recover your volume within a few minutes. You can use the Pause I/O action in AWS Fault Injection Service to run controlled experiments to test your architecture and monitoring based on this metric to improve your resiliency to storage faults.

Monitor I/O latency for a volume

You can monitor the average latency for read and write operations for an Amazon EBS volume using the `VolumeAvgReadLatency` and `VolumeAvgWriteLatency` metrics respectively. You can use the Latency Injection action in AWS Fault Injection Service to run controlled experiments to test your architecture and monitoring based on this metric to improve your resiliency to storage performance degradation.

If your I/O latency is higher than you require, make sure that your application is not attempting to drive more IOPS or throughput than you have provisioned for your volume. You can use the `VolumeAvgIOPS` and `VolumeAvgThroughput` metrics to monitor the average IOPS and throughput driven to your volume in a minute and then compare that with the volume's provisioned IOPS and throughput. If the volume does not drive any operations during the minute, the metrics will report a value of zero (`0`). If bursts of high IOPS or throughput occurred for a shorter time than the minute interval then the volume experiences micro-bursting, but the average IOPS and throughput metrics may report that you are driving lower performance than your volume's provisioned IOPS or througput limits. To identify whether your volume experiences performance bursts in a given minute, you can use the `VolumeIOPSExceededCheck` and `VolumeThroughputExceededCheck` metrics. You can monitor these metrics to determine whether your workload consistently attempted to drive IOPS or throughput that is greater than your volume's provisioned performance in a given minute. If the driven IOPS for any second within the minute consistently exceeds your volume's provisioned IOPS performance, the `VolumeIOPSExceededCheck` metric returns `1`. If the driven throughput for any second within the minute consistently exceeds your volume's provisioned throughput performance, the `VolumeThroughputExceededCheck` metric returns `1`. If driven IOPS and throughput is within your volume's provisioned performance, the metrics return `0`.

If your application requires a greater number of IOPS than your volume can provide, you should consider using one of the following:

- A `gp3`, `io2`, or `io1` volume that is provisioned with enough IOPS to achieve the required latency
- A larger `gp2` volume that provides enough baseline IOPS performance

HDD-backed `st1` and `sc1` volumes are designed to perform best with workloads that take advantage of the 1,024 KiB maximum I/O size. To determine your volume's average I/O size, divide `VolumeWriteBytes` by `VolumeWriteOps`. The same calculation applies to read operations. If average I/O size is below 64 KiB, increasing the size of the I/O operations sent to an `st1` or `sc1` volume should improve performance.

Monitor burst bucket balance for gp2, `st1`, and `sc1` volumes

`BurstBalance` displays the burst bucket balance for `gp2`, `st1`, and `sc1` volumes as a percentage of the remaining balance. When your burst bucket is depleted, volume I/O (for `gp2` volumes) or volume throughput (for `st1` and `sc1` volumes) is throttled to the baseline. Check the `BurstBalance` value to determine whether your volume is being throttled for this reason. For a complete list of the available Amazon EBS metrics, see Amazon CloudWatch metrics for Amazon EBS and Amazon EBS metrics for Nitro-based instances.

Monitor real-time I/O performance statistics

You can access real-time detailed performance statistics for Amazon EBS volumes that are attached to Nitro-based Amazon EC2 instances.

You can combine these statistics to derive average latency and IOPS, or to check whether I/O operations are completing. You can also view the total amount of time that your application has exceeded your EBS volume's or the attached instance's provisioned IOPS or throughput limits. By tracking increases in these statistics over time, you can identify whether you need to increase your provisioned IOPS or throughput limits to optimize your application's performance. The detailed performance statistics also include histograms for read and write I/O operations, which provide a distribution of your I/O latency by keeping track of the total number of I/O operations completed within a latency band.

For more information, see Amazon EBS detailed performance statistics.

Related resources

For more information about Amazon EBS I/O characteristics, see the following re:Invent presentation: Amazon EBS: Designing for Performance.

Amazon EBS and RAID configuration

With Amazon EBS, you can use any of the standard RAID configurations that you can use with a traditional bare metal server, as long as that particular RAID configuration is supported by the operating system for your instance. This is because all RAID is accomplished at the software level.

Amazon EBS volume data is replicated across multiple servers in an Availability Zone to prevent the loss of data from the failure of any single component. This replication makes Amazon EBS volumes ten times more reliable than typical commodity disk drives. For more information, see Amazon EBS features.

Contents

RAID configuration options

Creating a RAID 0 array allows you to achieve a higher level of performance for a file system than you can provision on a single Amazon EBS volume. Use RAID 0 when I/O performance is of the utmost importance. With RAID 0, I/O is distributed across the volumes in a stripe. If you add a volume, you get the straight addition of throughput and IOPS. However, keep in mind that performance of the stripe is limited to the worst performing volume in the set, and that the loss of a single volume in the set results in a complete data loss for the array.

The resulting size of a RAID 0 array is the sum of the sizes of the volumes within it, and the bandwidth is the sum of the available bandwidth of the volumes within it. For example, two 500 GiB `io1` volumes with 4,000 provisioned IOPS each create a 1,000 GiB RAID 0 array with an available bandwidth of 8,000 IOPS and 1,000 MiB/s of throughput.

> **Important**
>
> RAID 5 and RAID 6 are not recommended for Amazon EBS because the parity write operations of these RAID modes consume some of the IOPS available to your volumes. Depending on the configuration of your RAID array, these RAID modes provide 20-30% fewer usable IOPS than a RAID 0 configuration. Increased cost is a factor with these RAID modes as well; when using identical volume sizes and speeds, a 2-volume RAID 0 array can outperform a 4-volume RAID 6 array that costs twice as much.
> RAID 1 is also not recommended for use with Amazon EBS. RAID 1 requires more Amazon EC2 to Amazon EBS bandwidth than non-RAID configurations because the data is written to multiple volumes simultaneously. In addition, RAID 1 does not provide any write performance improvement.

Create a RAID 0 array

Use the following procedure to create the RAID 0 array.

Considerations

- Before you perform this procedure, you must decide how large your RAID 0 array should be and how many IOPS to provision.
- Create volumes with identical size and IOPS performance values for your array. Make sure you do not create an array that exceeds the available bandwidth of your EC2 instance.
- You should avoid booting from a RAID volume. If one of the devices fails, you might be unable to boot the operating system.

Linux instances

To create a RAID 0 array on Linux

1. Create the Amazon EBS volumes for your array. For more information, see Create an Amazon EBS volume.
2. Attach the Amazon EBS volumes to the instance that you want to host the array. For more information, see Attach an Amazon EBS volume to an Amazon EC2 instance.
3. Use the **mdadm** command to create a logical RAID device from the newly attached Amazon EBS volumes. Substitute the number of volumes in your array for `number_of_volumes` and the device names for each volume in the array (such as `/dev/xvdf`) for `device_name`. You can also substitute `MY_RAID` with your own unique name for the array.

   > **Note**
   >
   > You can list the devices on your instance with the **lsblk** command to find the device names.

   To create a RAID 0 array, run the following command (note the `--level=0` option to stripe the array):

   ```
   [ec2-user ~]$ sudo mdadm --create --verbose /dev/md0 --level=0 --name=MY_RAID --raid-devices=number_of_volumes device_name1 device_name2
   ```

> **Tip**
>
> If you get the `mdadm: command not found` error, use the following command to install mdadm: `sudo yum install mdadm`.

4. Allow time for the RAID array to initialize and synchronize. You can track the progress of these operations with the following command:

```
[ec2-user ~]$ sudo cat /proc/mdstat
```

The following is example output:

```
Personalities : [raid0]
md0 : active raid0 xvdc[1] xvdb[0]
      41910272 blocks super 1.2 512k chunks

unused devices: <none>
```

In general, you can display detailed information about your RAID array with the following command:

```
[ec2-user ~]$ sudo mdadm --detail /dev/md0
```

The following is example output:

```
/dev/md0:
           Version : 1.2
     Creation Time : Wed May 19 11:12:56 2021
        Raid Level : raid0
        Array Size : 41910272 (39.97 GiB 42.92 GB)
      Raid Devices : 2
     Total Devices : 2
       Persistence : Superblock is persistent

       Update Time : Wed May 19 11:12:56 2021
             State : clean
    Active Devices : 2
   Working Devices : 2
    Failed Devices : 0
```

```
      Spare Devices : 0

         Chunk Size : 512K

Consistency Policy : none

               Name : MY_RAID
               UUID : 646aa723:db31bbc7:13c43daf:d5c51e0c
             Events : 0

    Number   Major   Minor   RaidDevice State
       0      202      16        0      active sync   /dev/sdb
       1      202      32        1      active sync   /dev/sdc
```

5. Create a file system on your RAID array, and give that file system a label to use when you mount it later. For example, to create an ext4 file system with the label *MY_RAID*, run the following command:

   ```
   [ec2-user ~]$ sudo mkfs.ext4 -L MY_RAID /dev/md0
   ```

   Depending on the requirements of your application or the limitations of your operating system, you can use a different file system type, such as ext3 or XFS (consult your file system documentation for the corresponding file system creation command).

6. To ensure that the RAID array is reassembled automatically on boot, create a configuration file to contain the RAID information:

   ```
   [ec2-user ~]$ sudo mdadm --detail --scan | sudo tee -a /etc/mdadm.conf
   ```

   Note

   If you are using a Linux distribution other than Amazon Linux, you might need to modify this command. For example, you might need to place the file in a different location, or you might need to add the `--examine` parameter. For more information, run **man mdadm.conf** on your Linux instance.

7. Create a new ramdisk image to properly preload the block device modules for your new RAID configuration:

   ```
   [ec2-user ~]$ sudo dracut -H -f /boot/initramfs-$(uname -r).img $(uname -r)
   ```

8. Create a mount point for your RAID array.

```
[ec2-user ~]$ sudo mkdir -p /mnt/raid
```

9. Finally, mount the RAID device on the mount point that you created:

```
[ec2-user ~]$ sudo mount LABEL=MY_RAID /mnt/raid
```

   Your RAID device is now ready for use.

10. (Optional) To mount this Amazon EBS volume on every system reboot, add an entry for the device to the `/etc/fstab` file.

    a. Create a backup of your `/etc/fstab` file that you can use if you accidentally destroy or delete this file while you are editing it.

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

    b. Open the `/etc/fstab` file using your favorite text editor, such as **nano** or **vim**.

    c. Comment out any lines starting with "`UUID=`" and, at the end of the file, add a new line for your RAID volume using the following format:

    ```
    device_label mount_point file_system_type fs_mntops fs_freq fs_passno
    ```

    The last three fields on this line are the file system mount options, the dump frequency of the file system, and the order of file system checks done at boot time. If you don't know what these values should be, then use the values in the example below for them (`defaults,nofail 0 2`). For more information about `/etc/fstab` entries, see the **fstab** manual page (by entering **man fstab** on the command line). For example, to mount the ext4 file system on the device with the label MY_RAID at the mount point `/mnt/raid`, add the following entry to `/etc/fstab`.

    > **Note**
    >
    > If you ever intend to boot your instance without this volume attached (for example, so this volume could move back and forth between different instances), you should add the `nofail` mount option that allows the instance to boot even if

there are errors in mounting the volume. Debian derivatives, such as Ubuntu, must also add the `nobootwait` mount option.

```
LABEL=MY_RAID       /mnt/raid   ext4    defaults,nofail        0       2
```

d. After you've added the new entry to `/etc/fstab`, you need to check that your entry works. Run the **sudo mount -a** command to mount all file systems in `/etc/fstab`.

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

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

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

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

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

Windows instances

To create a RAID 0 array on Windows

1. Create the Amazon EBS volumes for your array. For more information, see Create an Amazon EBS volume.
2. Attach the Amazon EBS volumes to the instance that you want to host the array. For more information, see Attach an Amazon EBS volume to an Amazon EC2 instance.
3. Connect to your Windows instance. For more information, see Connect to your Windows instance.
4. Open a command prompt and type the **diskpart** command.

```
diskpart

Microsoft DiskPart version 6.1.7601
Copyright (C) 1999-2008 Microsoft Corporation.
On computer: WIN-BM6QPPL51CO
```

5. At the DISKPART prompt, list the available disks with the following command.

```
DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online           30 GB      0 B
  Disk 1    Online            8 GB      0 B
  Disk 2    Online            8 GB      0 B
```

   Identify the disks you want to use in your array and take note of their disk numbers.

6. Each disk you want to use in your array must be an online dynamic disk that does not contain any existing volumes. Use the following steps to convert basic disks to dynamic disks and to delete any existing volumes.

   a. Select a disk you want to use in your array with the following command, substituting *n* with your disk number.

   ```
   DISKPART> select disk n

   Disk n is now the selected disk.
   ```

   b. If the selected disk is listed as `Offline`, bring it online by running the **online disk** command.

   c. If the selected disk does not have an asterisk in the `Dyn` column in the previous **list disk** command output, you need to convert it to a dynamic disk.

   ```
   DISKPART> convert dynamic
   ```

> **Note**
>
> If you receive an error that the disk is write protected, you can clear the read-only flag with the **ATTRIBUTE DISK CLEAR READONLY** command and then try the dynamic disk conversion again.

d. Use the **detail disk** command to check for existing volumes on the selected disk.

```
DISKPART> detail disk

XENSRC PVDISK SCSI Disk Device
Disk ID: 2D8BF659
Type   : SCSI
Status : Online
Path   : 0
Target : 1
LUN ID : 0
Location Path : PCIROOT(0)#PCI(0300)#SCSI(P00T01L00)
Current Read-only State : No
Read-only  : No
Boot Disk  : No
Pagefile Disk  : No
Hibernation File Disk  : No
Crashdump Disk  : No
Clustered Disk  : No

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
  Volume 2     D   NEW VOLUME   FAT32  Simple      8189 MB  Healthy
```

Note any volume numbers on the disk. In this example, the volume number is 2. If there are no volumes, you can skip the next step.

e. (Only required if volumes were identified in the previous step) Select and delete any existing volumes on the disk that you identified in the previous step.

> **Warning**
>
> This destroys any existing data on the volume.

i. Select the volume, substituting *n* with your volume number.

```
DISKPART> select volume n
Volume n is the selected volume.
```

ii. Delete the volume.

```
DISKPART> delete volume

DiskPart successfully deleted the volume.
```

iii. Repeat these substeps for each volume you need to delete on the selected disk.

f. Repeat Step 6 for each disk you want to use in your array.

7. Verify that the disks you want to use are now dynamic. In this case, we're using disks 1 and 2 for the RAID volume.

```
DISKPART> list disk

  Disk ###  Status         Size     Free     Dyn  Gpt
  --------  -------------  -------  -------  ---  ---
  Disk 0    Online           30 GB      0 B
  Disk 1    Online            8 GB      0 B   *
  Disk 2    Online            8 GB      0 B   *
```

8. Create your raid array. On Windows, a RAID 0 volume is referred to as a striped volume.

   To create a striped volume array on disks 1 and 2, use the following command (note the `stripe` option to stripe the array):

```
DISKPART> create volume stripe disk=1,2
DiskPart successfully created the volume.
```

9. Verify your new volume.

```
DISKPART> list volume

  DISKPART> list volume

  Volume ###  Ltr  Label        Fs     Type        Size     Status     Info
  ----------  ---  -----------  -----  ----------  -------  ---------  --------
```

```
Volume 0     C                    NTFS   Partition      29 GB  Healthy    System
Volume 1                          RAW    Stripe         15 GB  Healthy
```

Note that the Type column now indicates that Volume 1 is a `stripe` volume.

10. Select and format your volume so that you can begin using it.

    a. Select the volume you want to format, substituting *n* with your volume number.

    ```
    DISKPART> select volume n

    Volume n is the selected volume.
    ```

    b. Format the volume.

    Note

    To perform a full format, omit the `quick` option.

    ```
    DISKPART> format quick recommended label="My new volume"

      100 percent completed

    DiskPart successfully formatted the volume.
    ```

    c. Assign an available drive letter to your volume.

    ```
    DISKPART> assign letter f

    DiskPart successfully assigned the drive letter or mount point.
    ```

Your new volume is now ready to use.

Create snapshots of volumes in a RAID array

If you want to back up the data on the EBS volumes in a RAID array using snapshots, you must ensure that the snapshots are consistent. This is because the snapshots of these volumes are created independently. To restore EBS volumes in a RAID array from snapshots that are out of sync would degrade the integrity of the array.

To create a consistent set of snapshots for your RAID array, use EBS multi-volume snapshots. Multi-volume snapshots allow you to take point-in-time, data coordinated, and crash-consistent snapshots across multiple EBS volumes attached to an EC2 instance. You do not have to stop your instance to coordinate between volumes to ensure consistency because snapshots are automatically taken across multiple EBS volumes. For more information, see the steps for creating multi-volume snapshots under Create Amazon EBS snapshots.

Benchmark Amazon EBS volumes

You can test the performance of Amazon EBS volumes by simulating I/O workloads. The process is as follows:

1. Launch an EBS-optimized instance.
2. Create new EBS volumes.
3. Attach the volumes to your EBS-optimized instance.
4. Configure and mount the block device.
5. Install a tool to benchmark I/O performance.
6. Benchmark the I/O performance of your volumes.
7. Delete your volumes and terminate your instance so that you don't continue to incur charges.

> **Important**
>
> Some of the procedures result in the destruction of existing data on the EBS volumes you benchmark. The benchmarking procedures are intended for use on volumes specially created for testing purposes, not production volumes.

Set up your instance

To get optimal performance from EBS volumes, we recommend that you use an EBS-optimized instance. EBS-optimized instances deliver dedicated throughput between Amazon EC2 and Amazon EBS, with instance. EBS-optimized instances deliver dedicated bandwidth between Amazon EC2 and Amazon EBS, with specifications depending on the instance type.

To create an EBS-optimized instance, choose **Launch as an EBS-optimized instance** when launching the instance using the Amazon EC2 console, or specify **--ebs-optimized** when using the command line. Be sure that you select an instance type that supports this option.

Set up Provisioned IOPS SSD or General Purpose SSD volumes

To create Provisioned IOPS SSD (`io1` and `io2`) or General Purpose SSD (`gp2` and `gp3`) volumes using the Amazon EC2 console, for **Volume type**, choose **Provisioned IOPS SSD (io1)**, **Provisioned IOPS SSD (io2)**, **General Purpose SSD (gp2)**, or **General Purpose SSD (gp3)**. At the command line, specify `io1`, `io2`, `gp2`, or `gp3` for the **--volume-type** parameter. For `io1`, `io2`, and `gp3` volumes, specify the number of I/O operations per second (IOPS) for the **--iops** parameter. For more information, see Amazon EBS volume types and Create an Amazon EBS volume.

(*Linux instances only*) For the example tests, we recommend that you create a RAID 0 array with 6 volumes, which offers a high level of performance. Because you are charged by gigabytes provisioned (and the number of provisioned IOPS for io1, io2, and gp3 volumes), not the number of volumes, there is no additional cost for creating multiple, smaller volumes and using them to create a stripe set. If you're using Oracle Orion to benchmark your volumes, it can simulate striping the same way that Oracle ASM does, so we recommend that you let Orion do the striping. If you are using a different benchmarking tool, you need to stripe the volumes yourself.

For more information about how to create a RAID 0 array, see Create a RAID 0 array.

Set up Throughput Optimized HDD (`st1`) or Cold HDD (`sc1`) volumes

To create an `st1` volume, choose **Throughput Optimized HDD** when creating the volume using the Amazon EC2 console, or specify **--type `st1`** when using the command line. To create an `sc1` volume, choose Cold HDD when creating the volume using the Amazon EC2 console, or specify **--type `sc1`** when using the command line. For information about creating EBS volumes, see Create an Amazon EBS volume. For information about attaching these volumes to your instance, see Attach an Amazon EBS volume to an Amazon EC2 instance.

(*Linux instances only*) AWS provides a JSON template for use with CloudFormation that simplifies this setup procedure. Access the template and save it as a JSON file. CloudFormation allows you to configure your own SSH keys and offers an easier way to set up a performance test environment to evaluate `st1` volumes. The template creates a current-generation instance and a 2 TiB `st1` volume, and attaches the volume to the instance at `/dev/xvdf`.

(*Linux instances only*) To create an HDD volume using the template

1. Open the CloudFormation console at https://console.aws.amazon.com/cloudformation.
2. Choose **Create Stack**.
3. Choose **Upload a Template to Amazon S3** and select the JSON template you previously obtained.
4. Give your stack a name like “ebs-perf-testing”, and select an instance type (the default is r3.8xlarge) and SSH key.
5. Choose **Next** twice, and then choose **Create Stack**.
6. After the status for your new stack moves from **CREATE_IN_PROGRESS** to **COMPLETE**, choose **Outputs** to get the public DNS entry for your new instance, which will have a 2 TiB `st1` volume attached to it.
7. Connect using SSH to your new stack as user **`ec2-user`**, with the hostname obtained from the DNS entry in the previous step.
8. Proceed to Install benchmark tools.

Install benchmark tools

The following tables lists some of the possible tools you can use to benchmark the performance of EBS volumes.

Linux instances

Tool	Description

fio	For benchmarking I/O performance. (Note that **fio** has a dependency on `libaio-devel` .)
	
	To install **fio** on Amazon Linux, run the following command:
	
	`$ sudo yum install -y fio`
	
	To install **fio** on Ubuntu, run the following command:
	
	`sudo apt-get install -y fio`

Tool	Description
Oracle Orion Calibration Tool	For calibrating the I/O performance of storage systems to be used with Oracle databases.

Windows instances

Tool	Description
DiskSpd	DiskSpd is a storage performance tool from the Windows, Windows Server, and Cloud Server Infrastructure engineering teams at Microsoft. It is available for download at https://github.com/Microsoft/diskspd/releases. After you download the `diskspd.exe` executable file, open a command prompt with administrative rights (by choosing "Run as Administrator"), and then navigate to the directory where you copied the `diskspd.exe` file. Copy the desired `diskspd.exe` executable file from the appropriate executable folder (`amd64fre`, `armfre` or `x86fre`) to a short, simple path like `C:\DiskSpd`. In most cases you will want the 64-bit version of DiskSpd from the `amd64fre` folder. The source code for DiskSpd is hosted on GitHub at: https://github.com/Microsoft/diskspd.
CrystalDiskMark	CrystalDiskMark is a simple disk benchmark software. It is available for download at https://crystalmark.info/en/software/crystaldiskmark/.

These benchmarking tools support a wide variety of test parameters. You should use commands that approximate the workloads your volumes will support. These commands provided below are intended as examples to help you get started.

Choose the volume queue length

Choosing the best volume queue length based on your workload and volume type.

Queue length on SSD-backed volumes

To determine the optimal queue length for your workload on SSD-backed volumes, we recommend that you target a queue length of 1 for every 1000 IOPS available (baseline for General Purpose SSD volumes and the provisioned amount for Provisioned IOPS SSD volumes). Then you can monitor your application performance and tune that value based on your application requirements.

Increasing the queue length is beneficial until you achieve the provisioned IOPS, throughput or optimal system queue length value, which is currently set to 32. For example, a volume with 3,000 provisioned IOPS should target a queue length of 3. You should experiment with tuning these values up or down to see what performs best for your application.

Queue length on HDD-backed volumes

To determine the optimal queue length for your workload on HDD-backed volumes, we recommend that you target a queue length of at least 4 while performing 1MiB sequential I/Os. Then you can monitor your application performance and tune that value based on your application requirements. For example, a 2 TiB `st1` volume with burst throughput of 500 MiB/s and IOPS of 500 should target a queue length of 4, 8, or 16 while performing 1,024 KiB, 512 KiB, or 256 KiB sequential I/Os respectively. You should experiment with tuning these values value up or down to see what performs best for your application.

Disable C-states

Before you run benchmarking, you should disable processor C-states. Temporarily idle cores in a supported CPU can enter a C-state to save power. When the core is called on to resume processing, a certain amount of time passes until the core is again fully operational. This latency can interfere with processor benchmarking routines. For more information about C-states and which EC2 instance types support them, see [Processor state control for your EC2 instance](#).

Linux instances

You can disable C-states on Amazon Linux, RHEL, and CentOS as follows:

1. Get the number of C-states.

```
$ cpupower idle-info | grep "Number of idle states:"
```

2. Disable the C-states from c1 to cN. Ideally, the cores should be in state c0.

```
$ for i in `seq 1 $((N-1))`; do cpupower idle-set -d $i; done
```

Windows instances

You can disable C-states on Windows as follows:

1. In PowerShell, get the current active power scheme.

```
$current_scheme = powercfg /getactivescheme
```

2. Get the power scheme GUID.

```
(Get-WmiObject -class Win32_PowerPlan -Namespace "root\cimv2\power" -Filter
 "ElementName='High performance'").InstanceID
```

3. Get the power setting GUID.

```
(Get-WmiObject -class Win32_PowerSetting -Namespace "root\cimv2\power" -Filter
 "ElementName='Processor idle disable'").InstanceID
```

4. Get the power setting subgroup GUID.

```
(Get-WmiObject -class Win32_PowerSettingSubgroup -Namespace "root\cimv2\power" -
Filter "ElementName='Processor power management'").InstanceID
```

5. Disable C-states by setting the value of the index to 1. A value of 0 indicates that C-states are disabled.

```
powercfg /
setacvalueindex <power_scheme_guid> <power_setting_subgroup_guid> <power_setting_guid>
 1
```

6. Set active scheme to ensure the settings are saved.

```
powercfg /setactive <power_scheme_guid>
```

Perform benchmarking

The following procedures describe benchmarking commands for various EBS volume types.

Run the following commands on an EBS-optimized instance with attached EBS volumes. If the EBS volumes were created from snapshots, be sure to initialize them before benchmarking. For more information, see Manually initialize the volumes after creation.

> **Tip**
>
> You can use the I/O latency histograms provided by the EBS detailed performance statistics to compare the distribution of I/O performance in your benchmarking tests. For more information, see Amazon EBS detailed performance statistics.

When you are finished testing your volumes, see the following topics for help cleaning up: Delete an Amazon EBS volume and Terminate your instance.

Benchmark Provisioned IOPS SSD and General Purpose SSD volumes

Linux instances

Run **fio** on the RAID 0 array that you created.

The following command performs 16 KB random write operations.

```
$ sudo fio --directory=/mnt/p_iops_vol0 --ioengine=psync --name fio_test_file --
direct=1 --rw=randwrite --bs=16k --size=1G --numjobs=16 --time_based --runtime=180 --
group_reporting --norandommap
```

The following command performs 16 KB random read operations.

```
$ sudo fio --directory=/mnt/p_iops_vol0 --name fio_test_file --direct=1 --rw=randread
 --bs=16k --size=1G --numjobs=16 --time_based --runtime=180 --group_reporting --
norandommap
```

For more information about interpreting the results, see this tutorial: Inspecting disk IO performance with fio.

Windows instances

Run **DiskSpd** on the volume that you created.

The following command will run a 30 second random I/O test using a 20GB test file located on the `C:` drive, with a 25% write and 75% read ratio, and an 8K block size. It will use eight worker threads, each with four outstanding I/Os, and a write entropy value seed of 1GB. The results of the test will be saved to a text file called `DiskSpeedResults.txt`. These parameters simulate a SQL Server OLTP workload.

```
diskspd -b8K -d30 -o4 -t8 -h -r -w25 -L -Z1G -c20G C:\iotest.dat > DiskSpeedResults.txt
```

For more information about interpreting the results, see this tutorial: [Inspecting disk IO performance with DiskSPd](#).

Benchmark `st1` and `sc1` volumes (Linux instances)

Run **fio** on your `st1` or `sc1` volume.

> **Note**
>
> Prior to running these tests, set buffered I/O on your instance as described in [Increase read-ahead for high-throughput, read-heavy workloads on `st1` and `sc1` (*Linux instances only*)](#).

The following command performs 1 MiB sequential read operations against an attached `st1` block device (for example, `/dev/xvdf`):

```
$ sudo fio --filename=/dev/<device> --direct=1 --rw=read --randrepeat=0
 --ioengine=libaio --bs=1024k --iodepth=8 --time_based=1 --runtime=180 --
name=fio_direct_read_test
```

The following command performs 1 MiB sequential write operations against an attached `st1` block device:

```
$ sudo fio --filename=/dev/<device> --direct=1 --rw=write --randrepeat=0
 --ioengine=libaio --bs=1024k --iodepth=8 --time_based=1 --runtime=180 --
name=fio_direct_write_test
```

Some workloads perform a mix of sequential reads and sequential writes to different parts of the block device. To benchmark such a workload, we recommend that you use separate, simultaneous **fio** jobs for reads and writes, and use the **fio** `offset_increment` option to target different block device locations for each job.

Running this workload is a bit more complicated than a sequential-write or sequential-read workload. Use a text editor to create a fio job file, called `fio_rw_mix.cfg` in this example, that contains the following:

```
[global]
clocksource=clock_gettime
randrepeat=0
runtime=180

[sequential-write]
bs=1M
ioengine=libaio
direct=1
iodepth=8
filename=/dev/<device>
do_verify=0
rw=write
rwmixread=0
rwmixwrite=100

[sequential-read]
bs=1M
ioengine=libaio
direct=1
iodepth=8
filename=/dev/<device>
do_verify=0
rw=read
rwmixread=100
rwmixwrite=0
offset=100g
```

Then run the following command:

```
$ sudo fio fio_rw_mix.cfg
```

For more information about interpreting the results, see this tutorial: [Inspecting disk I/O performance with fio](#).

Multiple **fio** jobs for direct I/O, even though using sequential read or write operations, can result in lower than expected throughput for `st1` and `sc1` volumes. We recommend that you use one direct I/O job and use the `iodepth` parameter to control the number of concurrent I/O operations.

Automate backups with Amazon Data Lifecycle Manager

You can use Amazon Data Lifecycle Manager to automate the creation, retention, and deletion of EBS snapshots and EBS-backed AMIs. When you automate snapshot and AMI management, it helps you to:

- Protect valuable data by enforcing a regular backup schedule.
- Create standardized AMIs that can be refreshed at regular intervals.
- Retain backups as required by auditors or internal compliance.
- Reduce storage costs by deleting outdated backups.
- Create disaster recovery backup policies that back up data to isolated Regions or accounts.

When combined with the monitoring features of Amazon EventBridge and AWS CloudTrail, Amazon Data Lifecycle Manager provides a complete backup solution for Amazon EC2 instances and individual EBS volumes at no additional cost.

Important

- Amazon Data Lifecycle Manager can't manage snapshots or AMIs created by any other means.
- Amazon Data Lifecycle Manager can't automate the creation, retention, and deletion of instance store-backed AMIs.

Amazon Data Lifecycle Manager is assessed as a service capability of Amazon Elastic Block Store (Amazon EBS). Any AWS Services in Scope by Compliance Program (FedRAMP, HIPAA BAA, SOC, etc) which lists Amazon EBS will also apply to Amazon Data Lifecycle Manager.

Contents

- Create Amazon Data Lifecycle Manager custom policy for EBS-backed AMIs
- Automate cross-account snapshot copies with Data Lifecycle Manager
- Modify Amazon Data Lifecycle Manager policies
- Delete Amazon Data Lifecycle Manager policies
- Control access to Amazon Data Lifecycle Manager using IAM
- Monitor Amazon Data Lifecycle Manager policies
- Service endpoints for Amazon Data Lifecycle Manager
- Create a private connection between a VPC and Amazon EBS
- Troubleshoot Amazon Data Lifecycle Manager issues

Quotas

Your AWS account has the following quotas related to Amazon Data Lifecycle Manager:

Description	Quota
Custom lifecycle policies per Region	100
Default policies for EBS snapshots per Region	1
Default policies for EBS-backed AMIs per Region	1
Tags per resource	45

How Amazon Data Lifecycle Manager works

The following are the key elements of Amazon Data Lifecycle Manager.

Elements

- Policies
- Policy schedules (custom policies only)

- Target resource tags (custom policies only)
- Snapshots
- EBS-backed AMIs
- Amazon Data Lifecycle Manager tags

Policies

With Amazon Data Lifecycle Manager, you create policies to define your backup creation and retention requirements. These policies typically specify the following:

- **Policy type** — Defines the type of backup resources that the policy manages (snapshots or EBS-backed AMIs).
- **Target resources** — Defines the type of resources that are targeted by the policy (instances or EBS volumes).
- **Creation frequency** — Defines how often the policy runs and creates snapshots or AMIs.
- **Retention threshold** — Defines how long the policy retains snapshots or AMIs after creation.
- **Additional actions** — Defines additional actions that the policy should perform, such as cross-Region copying, archiving, or resource tagging.

Amazon Data Lifecycle Manager offers default policies and custom policies.

Default policies

Default policies back up all volumes and instances in a Region that do not have recent backups. You can optionally exclude volumes and instances by specifying exclusion parameters.

Amazon Data Lifecycle Manager supports the following default policies:

- Default policy for EBS snapshots — Targets volumes and automates the creation, retention, and deletion of snapshots.
- Default policy for EBS-backed AMIs — Targets instances and automates the creation, retention, and deregistration of EBS-backed AMIs.

You can have only one default policy per resource type in each account and AWS Region.

Custom policies

Custom policies target specific resources based on their assigned tags and support advanced features, such as fast snapshot restore, snapshot archiving, cross-account copying, and pre and post scripts. A custom policy can include up to 4 schedules, where each schedule can have its own creation frequency, retention threshold, and advanced feature configuration.

Amazon Data Lifecycle Manager supports the following custom policies:

- EBS snapshot policy — Targets volumes or instances and automates the creation, retention, and deletion of EBS snapshots.
- EBS-backed AMI policy — Targets instances and automates the creation, retention, and deregistration of EBS-backed AMIs.
- Cross-account copy event policy — Automates cross-Region copy actions for snapshots that are shared with you.

For more information, see Amazon Data Lifecycle Manager default policies vs custom policies.

Policy schedules (*custom policies only*)

Policy schedules define when snapshots or AMIs are created by the policy. Policies can have up to four schedules—one mandatory schedule, and up to three optional schedules.

Adding multiple schedules to a single policy lets you create snapshots or AMIs at different frequencies using the same policy. For example, you can create a single policy that creates daily, weekly, monthly, and yearly snapshots. This eliminates the need to manage multiple policies.

For each schedule, you can define the frequency, fast snapshot restore settings (snapshot lifecycle policies only), cross-Region copy rules, and tags. The tags that are assigned to a schedule are automatically assigned to the snapshots or AMIs that are created when the schedule is initiated. In addition, Amazon Data Lifecycle Manager automatically assigns a system-generated tag based on the schedule's frequency to each snapshot or AMI.

Each schedule is initiated individually based on its frequency. If multiple schedules are initiated at the same time, Amazon Data Lifecycle Manager creates only one snapshot or AMI and applies the retention settings of the schedule that has the highest retention period. The tags of all of the initiated schedules are applied to the snapshot or AMI.

- (Snapshot lifecycle policies only) If more than one of the initiated schedules is enabled for fast snapshot restore, then the snapshot is enabled for fast snapshot restore in all of the Availability

Zones specified across all of the initiated schedules. The highest retention settings of the initiated schedules is used for each Availability Zone.

- If more than one of the initiated schedules is enabled for cross-Region copy, the snapshot or AMI is copied to all Regions specified across all of the initiated schedules. The highest retention period of the initiated schedules is applied.

Target resource tags (*custom policies only*)

Amazon Data Lifecycle Manager custom policies use resource tags to identify the resources to back up. When you create a snapshot or EBS-backed AMI policy, you can specify multiple target resource tags. All resources of the specified type (instance or volume) that have at least one of the specified target resource tags will be targeted by the policy. For example, if you create a snapshot policy that targets volumes and you specify `purpose=prod`, `costcenter=prod`, and `environment=live` as target resource tags, then the policy will target all volumes that have any of those tag-key value pairs.

If you want to run multiple policies on a resource, you can assign multiple tags to the target resource, and then create separate policies that each target a specific resource tag.

You can't use the \ or = characters in a tag key. Target resource tags are case sensitive. For more information, see Tag your resources.

Snapshots

Snapshots are the primary means to back up data from your EBS volumes. To save storage costs, successive snapshots are incremental, containing only the volume data that changed since the previous snapshot. When you delete one snapshot in a series of snapshots for a volume, only the data that's unique to that snapshot is removed. The rest of the captured history of the volume is preserved. For more information, see Amazon EBS snapshots.

EBS-backed AMIs

An Amazon Machine Image (AMI) provides the information that's required to launch an instance. You can launch multiple instances from a single AMI when you need multiple instances with the same configuration. Amazon Data Lifecycle Manager supports EBS-backed AMIs only. EBS-backed AMIs include a snapshot for each EBS volume that's attached to the source instance. For more information, see Amazon Machine Images (AMI).

Amazon Data Lifecycle Manager tags

Amazon Data Lifecycle Manager applies the following system tags to all snapshots and AMIs created by a policy, to distinguish them from snapshots and AMIs created by any other means:

- `aws:dlm:lifecycle-policy-id`
- `aws:dlm:lifecycle-schedule-name`
- `aws:dlm:expirationTime` — For snapshots created by an age-based schedule. Indicates when the snapshot is to be deleted from the standard tier.
- `dlm:managed`
- `aws:dlm:archived` — For snapshots that were archived by a schedule.
- `aws:dlm:pre-script` — For snapshots created with pre scripts.
- `aws:dlm:post-script` — For snapshots created with post scripts.

You can also specify custom tags to be applied to snapshots and AMIs on creation. You can't use the \ or = characters in a tag key.

The target tags that Amazon Data Lifecycle Manager uses to associate volumes with a snapshot policy can optionally be applied to snapshots created by the policy. Similarly, the target tags that are used to associate instances with an AMI policy can optionally be applied to AMIs created by the policy.

Amazon Data Lifecycle Manager default policies vs custom policies

This section compares default policies and custom policies and highlights their similarities and differences.

Topics

- EBS snapshot policy comparison
- EBS-backed AMI policy comparison

EBS snapshot policy comparison

The following table highlights the differences between the default policy for EBS snapshots and custom EBS snapshot policies.

Feature	Default policy for EBS snapshots	Custom EBS snapshot policy
Managed backup resource	EBS snapshot	EBS snapshot
Target resource types	Volumes	Volumes or instances
Resource targeting	Targets all volumes in the Region that do not have recent snapshots. You can specify exclusion parameter s to exclude specific volumes.	Targets only volumes or instances that have specific tags.
Exclusion parameters	Yes, can exclude boot volumes, specific volume types, and volumes with specific tags.	Yes, can exclude boot volumes and volumes with specific tags when targeting instances.
Support AWS Outposts	No	Yes
Support multiple schedules	No	Yes, up to 4 schedules per policy
Supported retention types	Age-based retention only	Age-based and count-based retention
Snapshot creation frequency	Every 1 to 7 days.	Daily, weekly, monthly, yearly, or custom frequency using a cron expression.
Snapshot retention	2 to 14 days.	Up to 1000 snapshots (count-based) or up to 100 years (age-based).

Feature	Default policy for EBS snapshots	Custom EBS snapshot policy
Support application-consistent snapshots	No	Yes, using pre and post scripts
Support snapshot archiving	No	Yes
Support fast snapshot restore	No	Yes
Support cross-Region copying	Yes, with default settings [1]	Yes, with custom settings
Support cross-account sharing	No	Yes
Support extended deletion [2]	Yes	No

[1] For default policies:

- You can't copy tags to cross-Region copies.
- Copies use the same retention period as the source snapshot.
- Copies get the same encryption state as the source snapshot. If the destination Region is enabled for encryption by default, copies are always encrypted, even if the source snapshots are unencrypted. Copies are always encrypted with the default KMS key for the destination Region.

[2] For default and custom policies:

- If a target instance or volume is deleted, Amazon Data Lifecycle Manager continues deleting snapshots up to, but not including, the last one based on the retention period. For default policies, you can extend deletion to include the last snapshot.
- If a policy is deleted or enters the error or disabled state, Amazon Data Lifecycle Manager stops deleting snapshots. For default policies, you can extend deletion to continue deleting snapshots, including the last one.

EBS-backed AMI policy comparison

The following table highlights the differences between the default policy for EBS-backed AMIs and custom EBS-backed AMI policies.

Feature	Default policy for EBS-backed AMIs	Custom EBS-backed AMI policy
Managed backup resource	EBS-backed AMIs	EBS-backed AMIs
Target resource types	Instances	Instances
Resource targeting	Targets all instances in the Region that do not have recent AMIs. You can specify exclusion parameters to exclude specific instances.	Targets only instances that have specific tags.
Reboot instances before AMI creation	No	Yes
Exclusion parameters	Yes, can exclude instances with specific tags.	No
Support multiple schedules	No	Yes, up to 4 schedules per policy.

Feature	Default policy for EBS-backed AMIs	Custom EBS-backed AMI policy
AMI creation frequency	Every 1 to 7 days.	Daily, weekly, monthly, yearly, or custom frequency using a cron expression.
Supported retention types	Age-based retention only.	Age-based and count-based retention.
AMIs retention	2 to 14 days.	Up to 1000 AMIs (count-based) or up to 100 years (age-based).
Support AMI deprecation	No	Yes
Support cross-Region copying	Yes, with default settings [1]	Yes, with custom settings
Support extended deletion [2]	Yes	No

[1]For default policies:

- You can't copy tags to cross-Region copies.
- Copies use the same retention period as the source AMI.
- Copies get the same encryption state as the source AMI. If the destination Region is enabled for encryption by default, copies are always encrypted, even if the source AMIs are unencrypted. Copies are always encrypted with the default KMS key for the destination Region.

[2] For default and custom policies:

- If a targeted instance is terminated, Amazon Data Lifecycle Manager continues deregistering AMIs up to, but not including, the last one based on the retention period. For default policies, you can extend deregistration to include the last AMI.

- If a policy is deleted or enters the error or disabled state, Amazon Data Lifecycle Manager stops deregistering AMIs. For default policies, you can extend deletion to continue deregistering AMIs, including the last one.

Create Amazon Data Lifecycle Manager default policies

To create periodic EBS-backed AMIs from instances, use the default policy for EBS-backed AMIs. To create snapshots of all volumes regardless of their attachment state, or if you want to exclude specific volumes, use the default policy for EBS snapshots.

This section explains how to create default policies.

Topics

Considerations for default policies

Keep the following in mind when working with default policies:

- Default policies do not back up target resources (instances or volumes) that have recent backups (snapshots or AMIs). The creation frequency determines which resources are backed up. A volume or instance is backed up only if its last snapshot or AMI is older than the policy's creation frequency. For example, if you specify a creation frequency of 3 days, the default policy for EBS snapshots will create a snapshot of a volume only if its last snapshot is older than 3 days.
- By default, default policies target all instances or volumes in the Region, unless exclusion parameters are specified.
- Default policies will create a minimum set of unique snapshots. For example, if you enable the EBS-backed AMI policy and the EBS snapshot policy, the snapshot policy will not duplicate snapshots of volumes that were already backed up by the EBS-backed AMI policy.
- Default policies will only start targeting resources that are at least 24 hours old.
- If you delete a volume or terminate an instance targeted by a default policy, Amazon Data Lifecycle Manager will continue to delete the previously created backups (snapshots or AMIs)

according to the retention period up to, but not including, the last backup. You must manually delete this backup if it is not required.

If you want Amazon Data Lifecycle Manager to delete the last backup, you can enable *extend deletion*.

- If a default policy is deleted or enters the error or disabled state, Amazon Data Lifecycle Manager stops deleting the previously created backups (snapshots or AMIs). If you want Amazon Data Lifecycle Manager to continue deleting backups, including the last one, you must enable *extend deletion* before deleting the policy or before the policy's state changes to disabled or deleted.
- When you create and enable a default policy, Amazon Data Lifecycle Manager randomly assigns targeted resources to a four-hour time window. Targeted resources are backed up during their assigned window at the specified creation frequency. For example, if a policy has a creation frequency of 3 days, and a target resource is assigned to the 12:00 - 16:00 window, that resource will be backed up between 12:00 - 16:00 every 3 days.

Create default policy for Amazon EBS snapshots

The following procedure shows you how to create a default policy for EBS snapshots.

Console

To create a default policy for EBS snapshots

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation panel, choose **Lifecycle Manager** and then choose **Create lifecycle policy**.
3. For **Policy type**, choose **Default policy** and then choose **EBS snapshot policy**.
4. For **Description**, enter a brief description for the policy.
5. For **IAM role**, choose the IAM role that has permissions to manage snapshots.

   We recommend that you choose **Default** to use the default IAM role provided by Amazon Data Lifecycle Manager. However, you can also use a custom IAM role that you previously created.
6. For **Creation frequency**, specify how often you want the policy to run and create snapshots of your volumes.

   The frequency that you specify also determines which volumes are backed up. The policy will only back up volumes that have not been backed up by any other means within the specified frequency. For example, if you specify a creation frequency of 3 days, the policy will only create snapshots of volumes that have not been backed up within the last 3 days.

7. For **Retention period**, specify how long you want the policy to retain the snapshots that it creates. When a snapshot reaches the retention threshold, it is automatically deleted. The retention period must be greater than or equal to the creation frequency.
8. (*Optional*) Configure the **Exclusion parameters** to exclude specific volumes from the scheduled backups. Excluded volumes will not be backed up when the policy runs.

    a. To exclude boot volumes, select **Exclude boot volumes**. If you exclude boot volumes, only data (non-boot) volumes will be backed up by the policy. In other words, it will not create snapshots of volumes that are attached to instances as a boot volume.

    b. To exclude specific volume types, choose **Exclude specific volume types**, and then select the volume types to exclude. Only volumes of the remaining types will be backed up by the policy.

    c. To exclude volumes that have specific tags, choose **Add tag**, and then specify the tag keys and values. The policy will not create snapshots of volumes that have any of the specified tags.
9. (*Optional*) In the **Advanced settings**, specify additional actions that the policy should perform.

    a. To copy assigned tags from the source volumes to their snapshots, select **Copy tags from volumes**.

    b. With **Extend deletion** disabled:

    - If a source volume is deleted, Amazon Data Lifecycle Manager continues to delete previously created snapshots up to, but not including, the last one based on the retention period. If you want Amazon Data Lifecycle Manager to delete all snapshots, including the last one, select **Extend deletion**.
    - If a policy is deleted or enters the `error` or `disabled` state, Amazon Data Lifecycle Manager stops deleting snapshots. If you want Amazon Data Lifecycle Manager to continue deleting snapshots, including the last one, select **Extend deletion**.

> **Note**
>
> If you enable extend deletion, you override both behaviors described above simultaneously.

c. To copy snapshots created by the policy to other Regions, select **Create cross-Region copy** and then select up to 3 destination Regions.

   - If the source snapshot is encrypted, or if encryption by default is enabled for the destination Region, the copied snapshots are encrypted using the default KMS key for EBS encryption in the destination Region.
   - If the source snapshot is unencrypted and encryption by default is disabled for the destination Region, the copied snapshots are unencrypted.

10. (*Optional*) To add a tag to the policy, choose **Add tag** and then specify the tag key and value pair.

11. Choose **Create default policy**.

> **Note**
>
> If you get the `Role with name AWSDataLifecycleManagerDefaultRole already exists` error, see Troubleshoot Amazon Data Lifecycle Manager issues for more information.

AWS CLI

To create a default policy for EBS snapshots

Use the create-lifecycle-policy command. You can specify the request parameters in one of two methods, depending on your use case or preferences:

- **Method 1**

```
$ aws dlm create-lifecycle-policy \
--state ENABLED | DISABLED \
--description "policy_description" \
--execution-role-arn role_arn \
```

```
--default-policy VOLUME \
--create-interval creation_frequency_in_days (1-7) \
--retain-interval retention_period_in_days (2-14) \
--copy-tags | --no-copy-tags \
--extend-deletion | --no-extend-deletion \
--cross-region-copy-targets TargetRegion=destination_region_code \
--exclusions ExcludeBootVolumes=true | false,
 ExcludeTags=[{Key=tag_key,Value=tag_value}], ExcludeVolumeTypes="standard | gp2 |
 gp3 | io1 | io2 | st1 | sc1"
```

For example, to create a default policy for EBS snapshots that targets all volumes in the Region, uses the default IAM role, runs daily (default), and retains snapshots for 7 days (default), you need to specify the following parameters:

```
$ aws dlm create-lifecycle-policy \
--state ENABLED \
--description "Daily default snapshot policy" \
--execution-role-arn arn:aws:iam::account_id:role/
AWSDataLifecycleManagerDefaultRole \
--default-policy VOLUME
```

- **Method 2**

```
$ aws dlm create-lifecycle-policy \
--state ENABLED | DISABLED \
--description "policy_description" \
--execution-role-arn role_arn \
--default-policy VOLUME \
--policy-details file://policyDetails.json
```

Where `policyDetails.json` includes the following:

```
{
    "PolicyLanguage": "SIMPLIFIED",
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceType": "VOLUME",
    "CopyTags": true | false,
    "CreateInterval": creation_frequency_in_days (1-7),
    "RetainInterval": retention_period_in_days (2-14),
    "ExtendDeletion": true | false,
    "CrossRegionCopyTargets": [{"TargetRegion":"destination_region_code"}],
    "Exclusions": {
```

```
        "ExcludeBootVolume": true | false,
  "ExcludeVolumeTypes": ["standard | gp2 | gp3 | io1 | io2 | st1 | sc1"],
        "ExcludeTags": [{
            "Key": "exclusion_tag_key",
            "Value": "exclusion_tag_value"
        }]
    }
}
```

Create default policy for EBS-backed AMIs

The following procedure shows you how to create a default policy for EBS-backed AMIs.

Console

To create a default policy for EBS-backed AMIs

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation panel, choose **Lifecycle Manager** and then choose **Create lifecycle policy**.
3. For **Policy type**, choose **Default policy** and then choose **EBS-backed AMI policy**.
4. For **Description**, enter a brief description for the policy.
5. For **IAM role**, choose the IAM role that has permissions to manage AMIs.

   We recommend that you choose **Default** to use the default IAM role provided by Amazon Data Lifecycle Manager. However, you can also use a custom IAM role that you previously created.
6. For **Creation frequency**, specify how often you want the policy to run and create AMIs from your instances.

   The frequency that you specify also determines which instances are backed up. The policy will only back up instances that have not been backed up by any other means within the specified frequency. For example, if you specify a creation frequency of 3 days, the policy will only create AMIs from instances that have not been backed up within the last 3 days.
7. For **Retention period**, specify how long you want the policy to retain the AMIs that it creates. When an AMI reaches the retention threshold, it is automatically deregistered and its associated snapshots are deleted. The retention period must be greater than or equal to the creation frequency.

8. (*Optional*) Configure the **Exclusion parameters** to exclude specific instances from the scheduled backups. Excluded instances will not be backed up when the policy runs.

   - To exclude instances that have specific tags, choose **Add tag**, and then specify the tag keys and values. The policy will not create AMIs from instances that have any of the specified tags.

9. (*Optional*) In the **Advanced settings**, specify additional actions that the policy should perform.

   a. To copy assigned tags from the source instances to their AMIs, select **Copy tags from instances**.

   b. With **Extend deletion** disabled:

      - If a source instance is terminated, Amazon Data Lifecycle Manager continues to deregister previously created AMIs up to, but not including, the last one based on the retention period. If you want Amazon Data Lifecycle Manager to deregister all AMIs, including the last one, select **Extend deletion**.
      - If a policy is deleted or enters the `error` or `disabled` state, Amazon Data Lifecycle Manager stops deregistering AMIs. If you want Amazon Data Lifecycle Manager to continue deregistering AMIs, including the last one, select **Extend deletion**.

   > **Note**
   >
   > If you enable extended deletion, you override both behaviors described above simultaneously.

   c. To copy AMIs created by the policy to other Regions, select **Create cross-Region copy** and then select up to 3 destination Regions.

      - If the source AMI is encrypted, or if encryption by default is enabled for the destination Region, the copied AMIs are encrypted using the default KMS key for EBS encryption in the destination Region.
      - If the source AMI is unencrypted and encryption by default is disabled for the destination Region, the copied AMIs are unencrypted.

10. (*Optional*) To add a tag to the policy, choose **Add tag** and then specify the tag key and value pair.

11. Choose **Create default policy**.

> **Note**
>
> If you get the `Role with name AWSDataLifecycleManagerDefaultRoleForAMIManagement already exists` error, see Troubleshoot Amazon Data Lifecycle Manager issues for more information.

AWS CLI

To create a default policy for EBS-backed AMIs

Use the create-lifecycle-policy command. You can specify the request parameters in one of two methods, depending on your use case or preferences:

- **Method 1**

```
$ aws dlm create-lifecycle-policy \
--state ENABLED | DISABLED \
--description "policy_description" \
--execution-role-arn role_arn \
--default-policy INSTANCE \
--create-interval creation_frequency_in_days (1-7) \
--retain-interval retention_period_in_days (2-14) \
--copy-tags | --no-copy-tags \
--extend-deletion | --no-extend-deletion \
--cross-region-copy-targets TargetRegion=destination_region_code \
--exclusions ExcludeTags=[{Key=tag_key,Value=tag_value}]
```

For example, to create a default policy for EBS-backed AMIs that targets all instances in the Region, uses the default IAM role, runs daily (default), and retains AMIs for 7 days (default), you need to specify the following parameters:

```
$ aws dlm create-lifecycle-policy \
--state ENABLED \
--description "Daily default AMI policy" \
--execution-role-arn arn:aws:iam::account_id:role/
AWSDataLifecycleManagerDefaultRoleForAMIManagement \
```

```
--default-policy INSTANCE
```

- **Method 2**

```
$ aws dlm create-lifecycle-policy \
--state ENABLED | DISABLED \
--description "policy_description" \
--execution-role-arn role_arn \
--default-policy INSTANCE \
--policy-details file://policyDetails.json
```

Where `policyDetails.json` includes the following:

```
{
    "PolicyLanguage": "SIMPLIFIED",
    "PolicyType": "IMAGE_MANAGEMENT",
    "ResourceType": "INSTANCE",
    "CopyTags": true | false,
    "CreateInterval": creation_frequency_in_days (1-7),
    "RetainInterval": retention_period_in_days (2-14),
    "ExtendDeletion": true | false,
 "CrossRegionCopyTargets": [{"TargetRegion":"destination_region_code"}],
    "Exclusions": {
        "ExcludeTags": [{
            "Key": "exclusion_tag_key",
            "Value": "exclusion_tag_value"
        }]
    }
}
```

Enable Data Lifecycle Manager default policies across accounts and Regions

Using CloudFormation StackSets, you can enable Amazon Data Lifecycle Manager default policies across multiple accounts and AWS Regions with a single operation.

You can use stack sets to enable default policies in one of the following ways:

- **Across an AWS organization** — Ensures that default policies are enabled and configured consistently across an entire AWS organization or specific organizational units in an organization.

This is done using *service-managed permissions*. CloudFormation StackSets creates the required IAM roles on your behalf.

- **Across specific AWS accounts** — Ensures that default policies are enabled and configured consistently across specific target accounts. This requires *self-managed permissions*. You create the IAM roles required to establish the trust relationship between the stack set administrator account and the target accounts.

For more information, see Permission models for stack sets in the *AWS CloudFormation User Guide*.

Use the following procedures to enable Amazon Data Lifecycle Manager default policies across an entire AWS organization, across specific OUs, or across specific target accounts.

Prerequisites

Do one of the following, depending on how you are enabling the default policies:

- (Across AWS organizations) You must enable all features in your organization and activate trusted access with AWS Organizations. You must also use the organization's management account or a delegated administrator account.
- (Across specific target accounts) You must grant self-managed permissions by creating the roles required to establish a trusted relationship between stack set administrator account and target accounts.

Console

To enable default policies across an AWS organization or across specific target accounts

1. Open the CloudFormation console at https://console.aws.amazon.com/cloudformation.
2. In the navigation pane, choose **StackSets**, then choose **Create StackSet**.
3. For **Permissions**, do one of the following, depending on how you are enabling the default policies:

    - (Across an AWS organization) Choose **Service-managed permissions**.
    - (Across specific target accounts) Choose **Self-service permissions**. Then, for **IAM admin role ARN**, select the IAM service role that that you created for the administrator account, and for **IAM execution role name**, enter the name of the IAM service role that you created in the target accounts.

4. For **Prepare template**, choose **Use a sample template**.
5. For **Sample templates**, do one of the following:
   - (Default policy for EBS snapshots) Select **Create Amazon Data Lifecycle Manager default policies for EBS Snapshots.**
   - (Default policy for EBS-backed AMIs) Select **Create Amazon Data Lifecycle Manager default policies for EBS-backed AMIs**.
6. Choose **Next**.
7. For **StackSet name** and **StackSet description**, enter a descriptive name and brief description.
8. In the **Parameters** section, configure the default policy settings as needed.

   > **Note**
   >
   > For critical workloads, we recommend **CreateInterval = 1 day** and **RetainInterval = 7 days**.

9. Choose **Next**.
10. (Optional) For **Tags**, specify tags to help you identify the StackSet and stack resources.
11. For **Managed execution**, choose **Active**.
12. Choose **Next**.
13. For **Add stacks to stack set**, choose **Deploy new stacks**.
14. Do one of the following, depending on how you are enabling the default policies:
    - (Across AWS organization) For **Deployment targets** choose one of the following options:
      - To deploy across an entire AWS organization, choose **Deploy to organization**.
      - To deploy to specific organizational units (OU), choose **Deploy to organizational units**, and then for **OU ID**, enter the OU ID. To add additional OUs, choose **Add another OU**.
    - (Across specific target accounts) For **Accounts**, do one of the following:
      - To deploy to specific target accounts, choose **Deploy stacks in accounts**, and then for **Account numbers**, enter the IDs of the target accounts.
      - To deploy to all accounts in a specific OU, choose **Deploy stack to all accounts in an organizational unit**, and then for **Organization numbers**, enter the ID of the target OU.

15. For **Automatic deployment**, choose **Activated**.
16. For **Account removal behavior**, choose **Retain stacks**.
17. For **Specify regions**, select specific Regions in which to enable default policies, or choose **Add all Regions** to enable default policies in all Regions.
18. Choose **Next**.
19. Review the stack set settings, select **I acknowledge that CloudFormation might create IAM resources**, and then choose **Submit**.

AWS CLI

To enable default policies across an AWS organization

1. Create the stack set. Use the create-stack-set command.

   For `--permission-model`, specify `SERVICE_MANAGED`.

   For `--template-url`, specify one of the following template URLs:

   - (Default policies for EBS-backed AMIs) `https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/DataLifecycleManagerAMIDefaultPolicy.yaml`
   - (Default policies for EBS snapshots) `https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/DataLifecycleManagerEBSSnapshotDefaultPolicy.yaml`

   For `--parameters`, specify the settings for the default policies. For supported parameters, parameter descriptions, and valid values, download the template using the URL and then view the template using a text editor.

   For `--auto-deployment`, specify `Enabled=true, RetainStacksOnAccountRemoval=true`.

```
$ aws cloudformation create-stack-set \
--stack-set-name stackset_name \
--permission-model SERVICE_MANAGED \
--template-url template_url \
--parameters "ParameterKey=param_name_1,ParameterValue=param_value_1"
 "ParameterKey=param_name_2,ParameterValue=param_value_2" \
```

```
--auto-deployment "Enabled=true, RetainStacksOnAccountRemoval=true"
```

2. Deploy the stack set. Use the [create-stack-instances] command.

   For `--stack-set-name`, specify the name of the stack set you created in the previous step.

   For `--deployment-targets OrganizationalUnitIds`, specify the ID of the root OU to deploy to an entire organization, or OU IDs to deploy to specific OUs in the organization.

   For `--regions`, specify the AWS Regions in which to enable the default policies.

```
$ aws cloudformation create-stack-instances \
--stack-set-name stackset_name \
--deployment-targets OrganizationalUnitIds='["root_ou_id"]' | '["ou_id_1",
 "ou_id_2]' \
--regions '["region_1", "region_2"]'
```

To enable default policies across specific target accounts

1. Create the stack set. Use the [create-stack-set] command.

   For `--template-url`, specify one of the following template URLs:

   - (Default policies for EBS-backed AMIs) `https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/DataLifecycleManagerAMIDefaultPolicy.yaml`
   - (Default policies for EBS snapshots) `https://s3.amazonaws.com/cloudformation-stackset-sample-templates-us-east-1/DataLifecycleManagerEBSSnapshotDefaultPolicy.yaml`

   For `--administration-role-arn`, specify the ARN of the IAM service role that you previously created for the stack set administrator.

   For `--execution-role-name`, specify the name of IAM service role that you created in the target accounts.

For `--parameters`, specify the settings for the default policies. For supported parameters, parameter descriptions, and valid values, download the template using the URL and then view the template using a text editor.

For `--auto-deployment`, specify `Enabled=true, RetainStacksOnAccountRemoval=true`.

```
$ aws cloudformation create-stack-set \
--stack-set-name stackset_name \
--template-url template_url \
--parameters "ParameterKey=param_name_1,ParameterValue=param_value_1"
 "ParameterKey=param_name_2,ParameterValue=param_value_2" \
--administration-role-arn administrator_role_arn \
--execution-role-name target_account_role \
--auto-deployment "Enabled=true, RetainStacksOnAccountRemoval=true"
```

2. Deploy the stack set. Use the create-stack-instances command.

   For `--stack-set-name`, specify the name of the stack set you created in the previous step.

   For `--accounts`, specify the IDs of the target AWS accounts.

   For `--regions`, specify the AWS Regions in which to enable the default policies.

```
$ aws cloudformation create-stack-instances \
--stack-set-name stackset_name \
--accounts '["account_ID_1","account_ID_2"]' \
--regions '["region_1", "region_2"]'
```

Create Amazon Data Lifecycle Manager custom policy for EBS snapshots

The following procedure shows you how to use Amazon Data Lifecycle Manager to automate Amazon EBS snapshot lifecycles.

Topics

- Considerations for snapshot lifecycle policies
- Additional resources
- Automate application-consistent snapshots with Data Lifecycle Manager
- Other use cases for Data Lifecycle Manager pre and post scripts
- How Amazon Data Lifecycle Manager pre and post scripts work
- Identify snapshots created with Data Lifecycle Manager pre and post scripts
- Monitor Amazon Data Lifecycle Manager pre and post scripts

Create a snapshot lifecycle policy

Use one of the following procedures to create a snapshot lifecycle policy.

Console

To create a snapshot policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**, and then choose **Create lifecycle policy**.
3. On the **Select policy type** screen, choose **EBS snapshot policy** and then choose **Next**.
4. In the **Target resources** section, do the following:

   a. For **Target resource types**, choose the type of resource to back up. Choose `Volume` to create snapshots of individual volumes, or choose `Instance` to create multi-volume snapshots from the volumes attached to an instance.

   b. (*Outpost and Local Zone customers only*) Specify where the target resources are located.

      For **Target resource location**, specify where the target resources are located.

      - To target resources in a Region, choose **AWS Region**. Amazon Data Lifecycle Manager will back up all resources of the specified type that have matching target tags in the current Region only. Snapshots are created in the same Region.
      - To target resources in Local Zones, choose **AWS Local Zones**. Amazon Data Lifecycle Manager will back up all resources of the specified type that have matching target

tags across all Local Zones in the current Region only. Snapshots can be created in the same Local Zone as the source resource, or in its parent Region.

- To target resources an Outpost, choose **AWS Outpost**. Amazon Data Lifecycle Manager will back up all resources of the specified type that have matching target tags across all Outposts in your account. Snapshots can be created on the same Outpost as the source resource, or in its parent Region.

c. For **Target resource tags**, choose the resource tags that identify the volumes or instances to back up. Only resources that have the specified tag key and value pairs are backed up by the policy.

5. For **Description**, enter a brief description for the policy.
6. For **IAM role**, choose the IAM role that has permissions to manage snapshots and to describe volumes and instances. To use the default role provided by Amazon Data Lifecycle Manager. choose **Default role**. Alternatively, to use a custom IAM role that you previously created, choose **Choose another role** and then select the role to use.
7. For **Policy tags**, add the tags to apply to the lifecycle policy. You can use these tags to identify and categorize your policies.
8. For **Policy status**, choose **Enable** to start the policy runs at the next scheduled time, or **Disable policy** to prevent the policy from running. If you do not enable the policy now, it will not start creating snapshots until you manually enable it after creation.
9. (*Policies that target instances only*) Exclude volumes from multi-volume snapshot sets.

   By default, Amazon Data Lifecycle Manager will create snapshots of all the volumes attached to targeted instances. However, you can choose to create snapshots of a subset of the attached volumes. In the **Parameters** section, do the following:

   - If you do not want to create snapshots of the root volumes attached to the targeted instances, select **Exclude root volume**. If you select this option, only the data (non-root) volumes that are attached to targeted instances will be included in the multi-volume snapshot sets.
   - If you want to create snapshots of a subset of the data (non-root) volumes attached to the targeted instances, select **Exclude specific data volumes**, and then specify the tags that are to be used to identify the data volumes that should not be snapshotted. Amazon Data Lifecycle Manager will not create snapshots of data volumes that have any of the specified tags. Amazon Data Lifecycle Manager will create snapshots only of data volumes that do not have any of the specified tags.

10. Choose **Next**.
11. On the **Configure schedule** screen, configure the policy schedules. A policy can have up to 4 schedules. Schedule 1 is mandatory. Schedules 2, 3, and 4 are optional. For each policy schedule that you add, do the following:

    a. In the **Schedule details** section do the following:

        i. For **Schedule name**, specify a descriptive name for the schedule.

        ii. For **Frequency** and the related fields, configure the interval between policy runs.

        You can configure policy runs on a daily, weekly, monthly, or yearly schedule. Alternatively, choose **Custom cron expression** to specify an interval of up to one year. For more information, see [Cron and rate expressions] in the *Amazon EventBridge User Guide*.

        > **Note**
        >
        > If you need to enable **snapshot archiving** for the schedule, then you must select either the **monthly** or **yearly** frequency, or you must specify a cron expression with a creation frequency of at least 28 days.
        > If specify a monthly frequency that creates snapshots on a specific day in a specific week (for example, the second Thursday of the month), then for count-based schedule, the retention count for the archive tier must be 4 or more.

        iii. For **Starting at**, specify the time at which the policy runs are scheduled to start. The first policy run starts within an hour after the scheduled time. The time must be entered in the `hh:mm` UTC format.

        iv. For **Retention type**, specify the retention policy for snapshots created by the schedule.

        You can retain snapshots based on either their total count or their age.

        - Count-based retention
            - With snapshot archiving disabled, the range is `1` to `1000`. When the retention threshold is reached, the oldest snapshot is permanently deleted.

  - With snapshot archiving enabled, the range is `0` (archive immediately after creation) to `1000`. When the retention threshold is reached, the oldest snapshot is converted to a full snapshot and it is moved to the archive tier.
- Age-based retention
  - With snapshot archiving disabled, the range is `1` day to `100` years. When the retention threshold is reached, the oldest snapshot is permanently deleted.
  - With snapshot archiving enabled, the range is `0` days (archive immediately after creation) to `100` years. When the retention threshold is reached, the oldest snapshot is converted to a full snapshot and it is moved to the archive tier.

> **Note**
>
> - All schedules must have the same retention type (age-based or count-based). You can specify the retention type for Schedule 1 only. Schedules 2, 3, and 4 inherit the retention type from Schedule 1. Each schedule can have its own retention count or period.
> - If you enable fast snapshot restore, cross-Region copy, or snapshot sharing, then you must specify a retention count of `1` or more, or a retention period of `1` day or longer.

v. (*AWS Outposts and Local Zone customers only*) Specify the snapshot destination.

For **Snapshot destination**, specify the destination for snapshots created by the policy.

- If the policy targets resources in a Region, snapshots must be created in the same Region. AWS Region is selected for you.
- If the policy targets resources in a Local Zone, you can create snapshots in the same Local Zone as the source resource, or in its parent Region.
- If the policy targets resources on an Outpost, you can create snapshots on the same Outpost as the source resource, or in its parent Region.

b. Configure tagging for snapshots.

In the **Tagging** section, do the following:

i. To copy all of the user-defined tags from the source volume to the snapshots created by the schedule, select **Copy tags from source**.

ii. To specify additional tags to assign to snapshots created by this schedule, choose **Add tags**.

c. Configure pre and post scripts for application-consistent snapshots.

For more information, see Automate application-consistent snapshots with Data Lifecycle Manager.

d. (*Policies that target volumes only*) Configure snapshot archiving.

In the **Snapshot archiving** section, do the following:

> **Note**
>
> You can enable snapshot archiving for only one schedule in a policy.

i. To enable snapshot archiving for the schedule, select **Archive snapshots created by this schedule**.

> **Note**
>
> You can enable snapshot archiving only if the snapshot creation frequency is monthly or yearly, or if you specify a cron expression with a creation frequency of at least 28 days.

ii. Specify the retention rule for snapshots in the archive tier.

- For **count-based schedules**, specify the number of snapshots to retain in the archive tier. When the retention threshold is reached, the oldest snapshot is permanently deleted from the archive tier. For example, if you specify 3, the schedule will retain a maximum of 3 snapshots in the archive tier. When the fourth snapshot is archived, the oldest of the three existing snapshots in the archive tier is deleted.
- For **age-based schedules**, specify the time period for which to retain snapshots in the archive tier. When the retention threshold is reached, the oldest snapshot is permanently deleted from the archive tier. For example, if you specify 120

days, the schedule will automatically delete snapshots from the archive tier when they reach that age.

> **Important**
>
> The minimum retention period for archived snapshots is 90 days. You must specify a retention rule that retains the snapshot for at least 90 days.

e. Enable fast snapshot restore.

To enable fast snapshot restore for snapshots created by the schedule, in the **Fast snapshot restore** section, select **Enable fast snapshot restore**. If you enable fast snapshot restore, you must choose the Availability Zones in which to enable it. If the schedule uses an age-based retention schedule, you must specify the period for which to enable fast snapshot restore for each snapshot. If the schedule uses count-based retention, you must specify the maximum number of snapshots to enable for fast snapshot restore.

If the schedule creates snapshots on an Outpost, you can't enable fast snapshot restore. Fast snapshot restore is not supported with local snapshots that are stored on an Outpost.

> **Note**
>
> You are billed for each minute that fast snapshot restore is enabled for a snapshot in a particular Availability Zone. Charges are pro-rated with a minimum of one hour.

f. Configure cross-Region copy.

To copy snapshots created by the schedule to an Outpost or to a different Region, in the **Cross-Region copy** section, select **Enable cross-Region copy**.

If the schedule creates snapshots in a Region, you can copy the snapshots to up to three additional Regions or Outposts in your account. You must specify a separate cross-Region copy rule for each destination Region or Outpost.

For each Region or Outpost, you can choose different retention policies and you can choose whether to copy all tags or no tags. If the source snapshot is encrypted, or if encryption by default is enabled, the copied snapshots are encrypted. If the source snapshot is unencrypted, you can enable encryption. If you do not specify a KMS key, the snapshots are encrypted using the default KMS key for EBS encryption in each destination Region. If you specify a KMS key for the destination Region, then the selected IAM role must have access to the KMS key.

> **Note**
>
> You must ensure that you do not exceed the number of concurrent snapshot copies per Region.

If the policy creates snapshots on an Outpost, then you can't copy the snapshots to a Region or to another Outpost and the cross-Region copy settings are not available.

g. Configure cross-account sharing.

   In the **Cross-account sharing**, configure the policy to automatically share the snapshots created by the schedule with other AWS accounts. Do the following:

   i. To enable sharing with other AWS accounts, select **Enable cross-account sharing**.

   ii. To add the accounts with which to share the snapshots, choose **Add account**, enter the 12-digit AWS account ID, and choose **Add**.

   iii. To automatically unshare shared snapshots after a specific period, select**Unshare automatically**. If you choose to automatically unshare shared snapshots, the period after which to automatically unshare the snapshots cannot be longer than the period for which the policy retains its snapshots. For example, if the policy's retention configuration retains snapshots for a period of 5 days, you can configure the policy to automatically unshare shared snapshots after periods up to 4 days. This applies to policies with age-based and count-based snapshot retention configurations.

      If you do not enable automatic unsharing, the snapshot is shared until it is deleted.

> **Note**
>
> You can only share snapshots that are unencrypted or that are encrypted using a customer managed key. You can't share snapshots that are encrypted with the default EBS encryption KMS key. If you share encrypted snapshots, then you must also share the KMS key that was used to encrypt the source volume with the target accounts. For more information, see Allowing users in other accounts to use a KMS key in the *AWS Key Management Service Developer Guide*.

h. To add additional schedules, choose **Add another schedule**, which is located at the top of the screen. For each additional schedule, complete the fields as described previously in this topic.

i. After you have added the required schedules, choose **Review policy**.

12. Review the policy summary, and then choose **Create policy**.

> **Note**
>
> If you get the `Role with name AWSDataLifecycleManagerDefaultRole already exists` error, see Troubleshoot Amazon Data Lifecycle Manager issues for more information.

Command line

Use the create-lifecycle-policy command to create a snapshot lifecycle policy. For `PolicyType`, specify `EBS_SNAPSHOT_MANAGEMENT`.

> **Note**
>
> To simplify the syntax, the following examples use a JSON file, `policyDetails.json`, that includes the policy details.

Example 1—Snapshot lifecycle policy with two schedules

This example creates a snapshot lifecycle policy that creates snapshots of all volumes that have a tag key of `costcenter` with a value of `115`. The policy includes two schedules. The first schedule creates a snapshot every day at 03:00 UTC. The second schedule creates a weekly snapshot every Friday at 17:00 UTC.

```
aws dlm create-lifecycle-policy \
    --description "My volume policy" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "VOLUME"
    ],
    "TargetTags": [{
        "Key": "costcenter",
        "Value": "115"
    }],
    "Schedules": [{
        "Name": "DailySnapshots",
        "TagsToAdd": [{
            "Key": "type",
            "Value": "myDailySnapshot"
        }],
        "CreateRule": {
            "Interval": 24,
            "IntervalUnit": "HOURS",
            "Times": [
                "03:00"
            ]
        },
        "RetainRule": {
            "Count": 5
        },
        "CopyTags": false
    },
    {
```

```
        "Name": "WeeklySnapshots",
        "TagsToAdd": [{
            "Key": "type",
            "Value": "myWeeklySnapshot"
        }],
        "CreateRule": {
            "CronExpression": "cron(0 17 ? * FRI *)"
        },
        "RetainRule": {
            "Count": 5
        },
        "CopyTags": false
    }
]}
```

If the request succeeds, the command returns the ID of the newly created policy. The following is example output.

```
{
   "PolicyId": "policy-0123456789abcdef0"
}
```

Example 2—Snapshot lifecycle policy that targets instances and creates snapshots of a subset of data (non-root) volumes

This example creates a snapshot lifecycle policy that creates multi-volume snapshot sets from instances tagged with `code=production`. The policy includes only one schedule. The schedule does not create snapshots of the data volumes that are tagged with `code=temp`.

```
aws dlm create-lifecycle-policy \
    --description "My volume policy" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
```

```
    ],
    "TargetTags": [{
        "Key": "code",
        "Value": "production"
    }],
    "Parameters": {
        "ExcludeDataVolumeTags": [{
            "Key": "code",
            "Value": "temp"
        }]
    },
    "Schedules": [{
        "Name": "DailySnapshots",
        "TagsToAdd": [{
            "Key": "type",
            "Value": "myDailySnapshot"
        }],
        "CreateRule": {
            "Interval": 24,
            "IntervalUnit": "HOURS",
            "Times": [
                "03:00"
            ]
        },
        "RetainRule": {
            "Count": 5
        },
        "CopyTags": false
    }
]}
```

If the request succeeds, the command returns the ID of the newly created policy. The following is example output.

```
{
   "PolicyId": "policy-0123456789abcdef0"
}
```

Example 3—Snapshot lifecycle policy that automates local snapshots of Outpost resources

This example creates a snapshot lifecycle policy that creates snapshots of volumes tagged with `team=dev` across all of your Outposts. The policy creates the snapshots on the same Outposts as the source volumes. The policy creates snapshots every 12 hours starting at `00:00` UTC.

```
aws dlm create-lifecycle-policy \
    --description "My local snapshot policy" \
    --state ENABLED \
    --execution-role-arn 
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": "VOLUME",
 "ResourceLocations": "OUTPOST",
    "TargetTags": [{
        "Key": "team",
        "Value": "dev"
    }],
    "Schedules": [{
        "Name": "on-site backup",
        "CreateRule": {
            "Interval": 12,
            "IntervalUnit": "HOURS",
            "Times": [
                "00:00"
            ],
 "Location": [
  "OUTPOST_LOCAL"
 ]
        },
        "RetainRule": {
            "Count": 1
        },
        "CopyTags": false
    }
]}
```

Example 4—Snapshot lifecycle policy that creates snapshots in a Region and copies them to an Outpost

The following example policy creates snapshots of volumes that are tagged with `team=dev`. Snapshots are created in the same Region as the source volume. Snapshots are created every 12 hours starting at `00:00` UTC, and retains a maximum of 1 snapshot. The policy also copies the snapshots to Outpost `arn:aws:outposts:us-east-1:123456789012:outpost/op-1234567890abcdef0`, encrypts the copied snapshots using the default encryption KMS key, and retains the copies for 1 month.

```
aws dlm create-lifecycle-policy \
    --description "Copy snapshots to Outpost" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": "VOLUME",
    "ResourceLocations": "CLOUD",
    "TargetTags": [{
        "Key": "team",
        "Value": "dev"
    }],
    "Schedules": [{
        "Name": "on-site backup",
        "CopyTags": false,
        "CreateRule": {
            "Interval": 12,
            "IntervalUnit": "HOURS",
            "Times": [
                "00:00"
            ],
            "Location": "CLOUD"
        },
        "RetainRule": {
            "Count": 1
        },
        "CrossRegionCopyRules" : [
        {
            "Target": "arn:aws:outposts:us-east-1:123456789012:outpost/
op-1234567890abcdef0",
            "Encrypted": true,
            "CopyTags": true,
            "RetainRule": {
                "Interval": 1,
```

```
                "IntervalUnit": "MONTHS"
            }
        }]
    }
]}
```

Example 5—Snapshot lifecycle policy with an archive-enabled, age-based schedule

This example creates a snapshot lifecycle policy that targets volumes tagged with `Name=Prod`. The policy has one age-based schedule that creates snapshots on the first day of each month at 09:00. The schedule retains each snapshot in the standard tier for one day, after which it moves them to the archive tier. Snapshots are stored in the archive tier for 90 days before being deleted.

```
aws dlm create-lifecycle-policy \
    --description "Copy snapshots to Outpost" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "ResourceTypes": [ "VOLUME"],
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "Schedules" : [
      {
        "Name": "sched1",
        "TagsToAdd": [
          {"Key":"createdby","Value":"dlm"}
        ],
        "CreateRule": {
          "CronExpression": "cron(0 9 1 * ? *)"
        },
        "CopyTags": true,
        "RetainRule":{
          "Interval": 1,
          "IntervalUnit": "DAYS"
        },
        "ArchiveRule": {
            "RetainRule":{
```

```
                "RetentionArchiveTier": {
                   "Interval": 90,
                   "IntervalUnit": "DAYS"
                }
              }
          }
        }
    ],
    "TargetTags": [
      {
        "Key": "Name",
        "Value": "Prod"
      }
    ]
}
```

Example 6—Snapshot lifecycle policy with an archive-enabled, count-based schedule

This example creates a snapshot lifecycle policy that targets volumes tagged with `Purpose=Test`. The policy has one count-based schedule that creates snapshots on the first day of each month at 09:00. The schedule archives snapshots immediately after creation and retains a maximum of three snapshots in the archive tier.

```
aws dlm create-lifecycle-policy \
    --description "Copy snapshots to Outpost" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "ResourceTypes": [ "VOLUME"],
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "Schedules" : [
      {
        "Name": "sched1",
        "TagsToAdd": [
          {"Key":"createdby","Value":"dlm"}
        ],
        "CreateRule": {
```

```
          "CronExpression": "cron(0 9 1 * ? *)"
        },
        "CopyTags": true,
        "RetainRule":{
          "Count": 0
        },
        "ArchiveRule": {
            "RetainRule":{
              "RetentionArchiveTier": {
                 "Count": 3
              }
            }
        }
      }
    ],
    "TargetTags": [
      {
        "Key": "Purpose",
        "Value": "Test"
      }
    ]
}
```

Considerations for snapshot lifecycle policies

The following **general considerations** apply to snapshot lifecycle policies:

- Snapshot lifecycle policies target only instances or volumes that are in the same Region as the policy.
- The first snapshot creation operation starts within one hour after the specified start time. Subsequent snapshot creation operations start within one hour of their scheduled time.
- You can create multiple policies to back up a volume or instance. For example, if a volume has two tags, where tag *A* is the target for policy *A* to create a snapshot every 12 hours, and tag *B* is the target for policy *B* to create a snapshot every 24 hours, Amazon Data Lifecycle Manager creates snapshots according to the schedules for both policies. Alternatively, you can achieve the same result by creating a single policy that has multiple schedules. For example, you can create a single policy that targets only tag *A*, and specify two schedules — one for every 12 hours and one for every 24 hours.
- Target resource tags are case sensitive.

- If you remove the target tags from a resource that is targeted by a policy, Amazon Data Lifecycle Manager no longer manages existing snapshots in the standard tier and archive tier; you must manually delete them if they are no longer needed.
- If you create a policy that targets instances, and new volumes are attached to a target instance after the policy has been created, the newly-added volumes are included in the backup at the next policy run. All volumes attached to the instance at the time of the policy run are included.
- If you create a policy with a custom cron-based schedule that is configured to create only one snapshot, the policy will not automatically delete that snapshot when the retention threshold is reached. You must manually delete the snapshot if it is no longer needed.
- If you create an age-based policy where the retention period is shorter than the creation frequency, Amazon Data Lifecycle Manager will always retain the last snapshot until the next one is created. For example, if an age-based policy creates one snapshot every month with a retention period of seven days, Amazon Data Lifecycle Manager will retain each snapshot for one month, even though the retention period is seven days.

The following considerations apply to **snapshot archiving**:

- You can enable snapshot archiving only for snapshot policies that target volumes.
- You can specify an archiving rule for only one schedule for each policy.
- If you are using the console, you can enable snapshot archiving only if the schedule has a monthly or yearly creation frequency, or if the schedule has a cron expression with a creation frequency of at least 28 days.

  If you are using the AWS CLI, AWS API, or AWS SDK, you can enable snapshot archiving only if the schedule has a cron expression with a creation frequency of at least 28 days.
- The minimum retention period in the archive tier is 90 days.
- When a snapshot is archived, it is converted to a full snapshot when it is moved to the archive tier. This could result in higher snapshot storage costs. For more information, see Pricing and billing for archiving Amazon EBS snapshots.
- Fast snapshot restore and snapshot sharing are disabled for snapshots when they are archived.
- If, in the case of a leap year, your retention rule results in an archive retention period of less than 90 days, Amazon Data Lifecycle Manager ensures that snapshots are retained for the minimum 90-day period.
- If you manually archive a snapshot created by Amazon Data Lifecycle Manager, and the snapshot is still archived when the schedule's retention threshold is reached, Amazon Data Lifecycle

Manager no longer manages that snapshot. However, if you restore the snapshot to the standard tier before the schedule's retention threshold is reached, the schedule will continue to manage the snapshot as per the retention rules.

- If you permanently or temporarily restore a snapshot archived by Amazon Data Lifecycle Manager to the standard tier, and the snapshot is still in the standard tier when the schedule's retention threshold is reached, Amazon Data Lifecycle Manager no longer manages the snapshot. However, if you re-archive the snapshot before the schedule's retention threshold is reached, the schedule will delete the snapshot when the retention threshold is met.
- Snapshots archived by Amazon Data Lifecycle Manager count towards your `Archived snapshots per volume` and `In-progress snapshot archives per account` quotas.
- If a schedule is unable to archive a snapshot after retrying for 24 hours, the snapshot remains in the standard tier and it is scheduled for deletion based on the time that it would have been deleted from the archive tier. For example, if the schedule archives snapshots for 120 days, it remains in the standard tier for 120 days after the failed archiving before being permanently deleted. For count-based schedules, the snapshot does not count towards the schedule's retention count.
- Snapshots must be archived in the same Region in which they were created. If you enabled cross-Region copy and snapshot archiving, Amazon Data Lifecycle Manager does not archive the snapshot copy.
- Snapshots archived by Amazon Data Lifecycle Manager are tagged with the `aws:dlm:archived=true` system tag. Additionally, snapshots created by an archive-enabled, age-based schedule are tagged with the `aws:dlm:expirationTime` system tag, which indicates the date and time at which the snapshot is scheduled to be archived.

The following considerations apply to **excluding root volumes and data (non-root) volumes**:

- If you choose to exclude boot volumes and you specify tags that consequently exclude all of the additional data volumes attached to an instance, then Amazon Data Lifecycle Manager will not create any snapshots for the affected instance, and it will emit a `SnapshotsCreateFailed` CloudWatch metric. For more information, see Monitor policies using CloudWatch.

The following considerations apply to **deleting volumes or terminating instances targeted by snapshot lifecycle policies**:

- If you delete a volume or terminate an instance targeted by a policy with a count-based retention schedule, Amazon Data Lifecycle Manager no longer manages snapshots in the standard tier and archive tier that were created from the deleted volume or instance. You must manually delete those earlier snapshots if they are no longer needed.
- If you delete a volume or terminate an instance targeted by a policy with an age-based retention schedule, the policy continues to delete snapshots from the standard tier and archive tier that were created from the deleted volume or instance on the defined schedule, up to, but not including, the last snapshot. You must manually delete the last snapshot if it is no longer needed.

The following considerations apply to snapshot lifecycle policies and **fast snapshot restore**:

- Amazon Data Lifecycle Manager can enable fast snapshot restore only for snapshots with a size of 16 TiB or less. For more information, see Amazon EBS fast snapshot restore.
- A snapshot that is enabled for fast snapshot restore remains enabled even if you delete or disable the policy, disable fast snapshot restore for the policy, or disable fast snapshot restore for the Availability Zone. You must disable fast snapshot restore for these snapshots manually.
- If you enable fast snapshot restore for a policy and you exceed the maximum number of snapshots that can be enabled for fast snapshot restore, Amazon Data Lifecycle Manager creates snapshots as scheduled but does not enable them for fast snapshot restore. After a snapshot that is enabled for fast snapshot restore is deleted, the next snapshot that Amazon Data Lifecycle Manager creates is enabled for fast snapshot restore.
- When fast snapshot restore is enabled for a snapshot, it takes 60 minutes per TiB to optimize the snapshot. We recommend that you configure your schedules so that each snapshot is fully optimized before Amazon Data Lifecycle Manager creates the next snapshot.
- If you enable fast snapshot restore for a policy that targets instances, Amazon Data Lifecycle Manager enables fast snapshot restore for each snapshot in the multi-volume snapshot set individually. If Amazon Data Lifecycle Manager fails to enable fast snapshot restore for one of the snapshots in the multi-volume snapshot set, it will still attempt to enable fast snapshot restore for the remaining snapshots in the snapshot set.
- You are billed for each minute that fast snapshot restore is enabled for a snapshot in a particular Availability Zone. Charges are pro-rated with a minimum of one hour. For more information, see Pricing and Billing.

> **Note**
>
> Depending on the configuration of your lifecycle policies, you could have multiple snapshots enabled for fast snapshot restore in multiple Availability Zones simultaneously.

The following considerations apply to snapshot lifecycle policies and **Multi-Attach enabled volumes**:

- When creating a lifecycle policy that targets instances that have the same Multi-Attach enabled volume, Amazon Data Lifecycle Manager initiates a snapshot of the volume for each attached instance. Use the *timestamp* tag to identify the set of time-consistent snapshots that are created from the attached instances.

The following considerations apply to **sharing snapshots across accounts**:

- You can only share snapshots that are unencrypted or that are encrypted using a customer managed key.
- You can't share snapshots that are encrypted with the default EBS encryption KMS key.
- If you share encrypted snapshots, you must also share the KMS key that was used to encrypt the source volume with the target accounts. For more information, see Allowing users in other accounts to use a KMS key in the *AWS Key Management Service Developer Guide*.

The following considerations apply to snapshots policies and **snapshot archiving**:

- If you manually archive a snapshot that was created by a policy, and that snapshot is in the archive tier when the policy's retention threshold is reached, Amazon Data Lifecycle Manager will not delete the snapshot. Amazon Data Lifecycle Manager does not manage snapshots while they are stored in the archive tier. If you no longer need snapshots that are stored in the archive tier, you must manually delete them.

The following considerations apply to snapshot policies and Recycle Bin:

- If Amazon Data Lifecycle Manager deletes a snapshot and sends it to the Recycle Bin when the policy's retention threshold is reached, and you manually restore the snapshot from the Recycle

Bin, you must manually delete that snapshot when it is no longer needed. Amazon Data Lifecycle Manager will no longer manage the snapshot.

- If you manually delete a snapshot that was created by a policy, and that snapshot is in the Recycle Bin when the policy's retention threshold is reached, Amazon Data Lifecycle Manager will not delete the snapshot. Amazon Data Lifecycle Manager does not manage the snapshots while they are stored in the Recycle Bin.

  If the snapshot is restored from the Recycle Bin before the policy's retention threshold is reached, Amazon Data Lifecycle Manager will delete the snapshot when the policy's retention threshold is reached.

  If the snapshot is restored from the Recycle Bin after the policy's retention threshold is reached, Amazon Data Lifecycle Manager will no longer delete the snapshot. You must manually delete the snapshot when it is no longer needed.

The following considerations apply to snapshot lifecycle policies that are in the **error** state:

- For policies with age-based retention schedules, snapshots that are set to expire while the policy is in the `error` state are retained indefinitely. You must delete the snapshots manually. When you re-enable the policy, Amazon Data Lifecycle Manager resumes deleting snapshots as their retention periods expire.
- For policies with count-based retention schedules, the policy stops creating and deleting snapshots while it is in the `error` state. When you re-enable the policy, Amazon Data Lifecycle Manager resumes creating snapshots, and it resumes deleting snapshots as the retention threshold is met.

The following considerations apply to snapshot policies and **snapshot lock**:

- If you manually lock a snapshot created by Amazon Data Lifecycle Manager, and that snapshot is still locked when its retention threshold is reached, Amazon Data Lifecycle Manager no longer manages that snapshot. You must manually delete the snapshot if it is no longer needed.
- If you manually lock a snapshot that was created and enabled for fast snapshot restore by Amazon Data Lifecycle Manager, and the snapshot is still locked when its retention threshold is reached, Amazon Data Lifecycle Manager will not disable fast snapshot restore or delete the snapshot. You must manually disable fast snapshot restore and delete the snapshot if it is no longer needed.

- If you manually register a snapshot that was created by Amazon Data Lifecycle Manager with an AMI and then lock that snapshot, and that snapshot is still locked and associated with the AMI when its retention threshold is reached, Amazon Data Lifecycle Manager will continue to attempt to delete that snapshot. When the AMI is deregistered and the snapshot is unlocked, Amazon Data Lifecycle Manager will automatically delete the snapshot.

Additional resources

For more information, see the Automating Amazon EBS snapshot and AMI management using Amazon Data Lifecycle Manager AWS storage blog.

Automate application-consistent snapshots with Data Lifecycle Manager

You can automate application-consistent snapshots with Amazon Data Lifecycle Manager by enabling pre and post scripts in your snapshot lifecycle policies that target instances.

Amazon Data Lifecycle Manager integrates with AWS Systems Manager (Systems Manager) to support application-consistent snapshots. Amazon Data Lifecycle Manager uses Systems Manager (SSM) command documents that include pre and post scripts to automate the actions needed to complete application-consistent snapshots. Before Amazon Data Lifecycle Manager initiates snapshot creation, it runs the commands in the pre script to freeze and flush I/O. After Amazon Data Lifecycle Manager initiates snapshot creation, it runs the commands in the post script to thaw I/O.

Using Amazon Data Lifecycle Manager, you can automate application-consistent snapshots of the following:

- Windows applications using Volume Shadow Copy Service (VSS)
- SAP HANA using an AWS managed SSDM document. For more information, see Amazon EBS snapshots for SAP HANA.
- Self-managed databases, such as MySQL, PostgreSQL or InterSystems IRIS, using SSM document templates

Topics

- Requirements for using pre and post scripts
- Getting started with application-consistent snapshots

- Considerations for VSS Backups with Amazon Data Lifecycle Manager
- Shared responsibility for application-consistent snapshots

Requirements for using pre and post scripts

The following table outlines the requirements for using pre and post scripts with Amazon Data Lifecycle Manager.

	Application-consistent snapshots		
Requirement	VSS Backup	Custom SSM document	Other use cases
SSM Agent installed and running on target instances	✓	✓	✓
VSS system requirements met on target instances	✓		
VSS enabled instance profile associated with target instances	✓		
VSS components installed on target instances	✓		
Prepare SSM document with pre and post script commands		✓	✓
Prepare Amazon Data Lifecycle Manager IAM role run pre and post scripts	✓	✓	✓

	Application-consistent snapshots		
Create snapshot policy that targets instances and is configured for pre and post scripts	✓	✓	✓

Getting started with application-consistent snapshots

This section explains the steps you need to follow to automate application-consistent snapshots using Amazon Data Lifecycle Manager.

Step 1: Prepare target instances

You need to prepare the targeted instances for application-consistent snapshots using Amazon Data Lifecycle Manager. Do one of the following, depending on your use case.

Prepare for VSS Backups

To prepare your target instances for VSS backups

1. Install the SSM Agent on your target instances, if it is not already installed. If SSM Agent is already installed on your target instances, skip this step.

   For more information, see Working with SSM Agent on EC2 instances for Windows server.
2. Ensure that the SSM Agent is running. For more information, see Checking SSM Agent status and starting the agent.
3. Set up Systems Manager for Amazon EC2 instances. For more information, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.
4. Ensure the system requirements for VSS backups are met.
5. Attach a VSS-enabled instance profile to the target instances.
6. Install the VSS components.

Prepare for SAP HANA backups

To prepare your target instances for SAP HANA backups

1. Prepare the SAP HANA environment on your target instances.

   a. Set up your instance with SAP HANA. If you don't already have an existing SAP HANA environment, then you can refer to the [SAP HANA Environment Setup on AWS](#).

   b. Login to the SystemDB as a suitable administrator user.

   c. Create a database backup user to be used with Amazon Data Lifecycle Manager.

   ```
   CREATE USER username PASSWORD password NO FORCE_FIRST_PASSWORD_CHANGE;
   ```

   For example, the following command creates a user named `dlm_user` with password `password`.

   ```
   CREATE USER dlm_user PASSWORD password NO FORCE_FIRST_PASSWORD_CHANGE;
   ```

   d. Assign the `BACKUP OPERATOR` role to the database backup user that you created in the previous step.

   ```
   GRANT BACKUP OPERATOR TO username
   ```

   For example, the following command assigns the role to a user named `dlm_user`.

   ```
   GRANT BACKUP OPERATOR TO dlm_user
   ```

   e. Log in to the operating system as the administrator, for example *`sid`*`adm`.

   f. Create an `hdbuserstore` entry to store connection information so that the SAP HANA SSM document can connect to SAP HANA without users having to enter the information.

   ```
   hdbuserstore set DLM_HANADB_SNAPSHOT_USER
    localhost:3hana_instance_number13 username password
   ```

   For example:

   ```
   hdbuserstore set DLM_HANADB_SNAPSHOT_USER localhost:30013 dlm_user password
   ```

g. Test the connection.

```
hdbsql -U DLM_HANADB_SNAPSHOT_USER "select * from dummy"
```

2. Install the SSM Agent on your target instances, if it is not already installed. If SSM Agent is already installed on your target instances, skip this step.

   For more information, see Manually installing SSM Agent on EC2 instances for Linux.

3. Ensure that the SSM Agent is running. For more information, see Checking SSM Agent status and starting the agent.

4. Set up Systems Manager for Amazon EC2 instances. For more information, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.

Prepare for custom SSM documents

To prepare your target instances custom SSM documents

1. Install the SSM Agent on your target instances, if it is not already installed. If SSM Agent is already installed on your target instances, skip this step.
   - (Linux instances) Manually installing SSM Agent on EC2 instances for Linux
   - (Windows instances) Working with SSM Agent on EC2 instances for Windows Server
2. Ensure that the SSM Agent is running. For more information, see Checking SSM Agent status and starting the agent.
3. Set up Systems Manager for Amazon EC2 instances. For more information, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.

Step 2: Prepare SSM document

> **Note**
>
> This step is required only for custom SSM documents. It is not required for VSS Backup or SAP HANA. For VSS Backups and SAP HANA, Amazon Data Lifecycle Manager uses the AWS managed SSM document.

If you are automating application-consistent snapshots for a self-managed database, such as MySQL, PostgreSQL, or InterSystems IRIS, you must create an SSM command document that includes a pre script to freeze and flush I/O before snapshot creation is initiated, and a post script to thaw I/O after snapshot creation is initiated.

If your MySQL, PostgreSQL, or InterSystems IRIS database uses standard configurations, you can create an SSM command document using the sample SSM document content below. If your MySQL, PostgreSQL, or InterSystems IRIS database uses a non-standard configuration, you can use the sample content below as a starting point for your SSM command document and then customize it to meet your requirements. Alternatively, if you want to create a new SSM document from scratch, you can use the empty SSM document template below and add your pre and post commands in the appropriate document sections.

Note the following:

- It is your responsibility to ensure that the SSM document performs the correct and required actions for your database configuration.
- Snapshots are guaranteed to be application-consistent only if the pre and post scripts in your SSM document can successfully freeze, flush, and thaw I/O.
- The SSM document must include required fields for `allowedValues`, including `pre-script`, `post-script`, and `dry-run`. Amazon Data Lifecycle Manager will execute commands on your instance based on the contents of those sections. If your SSM document does not have those sections, then Amazon Data Lifecycle Manager will treat it as a failed execution.

MySQL sample document content

```
###===============================================================================###
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.
```

```
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
###===============================================================================###
schemaVersion: '2.2'
description: Amazon Data Lifecycle Manager Pre/Post script for MySQL databases
parameters:
  executionId:
    type: String
    default: None
    description: (Required) Specifies the unique identifier associated with a pre
 and/or post execution
    allowedPattern: ^(None|[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]
{4}-[a-fA-F0-9]{12})$
  command:
  # Data Lifecycle Manager will trigger the pre-script and post-script actions
 during policy execution.
  # 'dry-run' option is intended for validating the document execution without
 triggering any commands
  # on the instance. The following allowedValues will allow Data Lifecycle Manager
 to successfully
  # trigger pre and post script actions.
    type: String
    default: 'dry-run'
    description: (Required) Specifies whether pre-script and/or post-script should
 be executed.
    allowedValues:
    - pre-script
    - post-script
    - dry-run

mainSteps:
- action: aws:runShellScript
  description: Run MySQL Database freeze/thaw commands
  name: run_pre_post_scripts
  precondition:
    StringEquals:
    - platformType
    - Linux
  inputs:
```

```
    runCommand:
    - |
      #!/bin/bash


###===============================================================================###
      ### Error Codes

###===============================================================================###
      # The following Error codes will inform Data Lifecycle Manager of the type of
error
      # and help guide handling of the error.
      # The Error code will also be emitted via AWS Eventbridge events in the
'cause' field.
      # 1 Pre-script failed during execution - 201
      # 2 Post-script failed during execution - 202
      # 3 Auto thaw occurred before post-script was initiated - 203
      # 4 Pre-script initiated while post-script was expected - 204
      # 5 Post-script initiated while pre-script was expected - 205
      # 6 Application not ready for pre or post-script initiation - 206

      ###=================================================================###
      ### Global variables
      ###=================================================================###
      START=$(date +%s)
      # For testing this script locally, replace the below with OPERATION=$1.
      OPERATION={{ command }}
      FS_ALREADY_FROZEN_ERROR='freeze failed: Device or resource busy'
      FS_ALREADY_THAWED_ERROR='unfreeze failed: Invalid argument'
      FS_BUSY_ERROR='mount point is busy'

      # Auto thaw is a fail safe mechanism to automatically unfreeze the application
after the
      # duration specified in the global variable below. Choose the duration based
on your
      # database application's tolerance to freeze.
      export AUTO_THAW_DURATION_SECS="60"

      # Add all pre-script actions to be performed within the function below
      execute_pre_script() {
          echo "INFO: Start execution of pre-script"
          # Check if filesystem is already frozen. No error code indicates that
filesystem
```

```
        # is not currently frozen and that the pre-script can proceed with
freezing the filesystem.
        check_fs_freeze
        # Execute the DB commands to flush the DB in preparation for snapshot
        snap_db
        # Freeze the filesystem. No error code indicates that filesystem was
succefully frozen
        freeze_fs

        echo "INFO: Schedule Auto Thaw to execute in ${AUTO_THAW_DURATION_SECS}
seconds."
        $(nohup bash -c execute_schedule_auto_thaw  >/dev/null 2>&1 &)
    }

    # Add all post-script actions to be performed within the function below
    execute_post_script() {
        echo "INFO: Start execution of post-script"
        # Unfreeze the filesystem. No error code indicates that filesystem was
successfully unfrozen.
        unfreeze_fs
        thaw_db
    }

    # Execute Auto Thaw to automatically unfreeze the application after the
duration configured
    # in the AUTO_THAW_DURATION_SECS global variable.
    execute_schedule_auto_thaw() {
        sleep ${AUTO_THAW_DURATION_SECS}
        execute_post_script
    }

    # Disable Auto Thaw if it is still enabled
    execute_disable_auto_thaw() {
        echo "INFO: Attempting to disable auto thaw if enabled"
        auto_thaw_pgid=$(pgrep -f execute_schedule_auto_thaw | xargs -i ps -hp {}
-o pgid)
        if [ -n "${auto_thaw_pgid}" ]; then
            echo "INFO: execute_schedule_auto_thaw process found with pgid
${auto_thaw_pgid}"
            sudo pkill -g ${auto_thaw_pgid}
            rc=$?
            if [ ${rc} != 0 ]; then
                echo "ERROR: Unable to kill execute_schedule_auto_thaw process.
retval=${rc}"
```

```
            else
                echo "INFO: Auto Thaw  has been disabled"
            fi
        fi
    }

    # Iterate over all the mountpoints and check if filesystem is already in
freeze state.
    # Return error code 204 if any of the mount points are already frozen.
    check_fs_freeze() {
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
            # Freeze of the root and boot filesystems is dangerous and pre-script
does not freeze these filesystems.
            # Hence, we will skip the root and boot mountpoints while checking if
filesystem is in freeze state.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi

            error_message=$(sudo mount -o remount,noatime $target 2>&1)
            # Remount will be a no-op without a error message if the filesystem is
unfrozen.
            # However, if filesystem is already frozen, remount will fail with
busy error message.
            if [ $? -ne 0 ];then
                # If the filesystem is already in frozen, return error code 204
                if [[ "$error_message" == *"$FS_BUSY_ERROR"* ]];then
                    echo "ERROR: Filesystem ${target} already frozen. Return Error
Code: 204"
                    exit 204
                fi
                # If the check filesystem freeze failed due to any reason other
than the filesystem already frozen, return 201
                echo "ERROR: Failed to check_fs_freeze on mountpoint $target due
to error - $errormessage"
                exit 201
            fi
        done
    }

    # Iterate over all the mountpoints and freeze the filesystem.
    freeze_fs() {
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
```

```
            # Freeze of the root and boot filesystems is dangerous. Hence, skip
filesystem freeze
            # operations for root and boot mountpoints.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi
            echo "INFO: Freezing $target"
            error_message=$(sudo fsfreeze -f $target 2>&1)
            if [ $? -ne 0 ];then
                # If the filesystem is already in frozen, return error code 204
                if [[ "$error_message" == *"$FS_ALREADY_FROZEN_ERROR"* ]]; then
                    echo "ERROR: Filesystem ${target} already frozen. Return Error
Code: 204"
                    sudo mysql -e 'UNLOCK TABLES;'
                    exit 204
                fi
                # If the filesystem freeze failed due to any reason other than the
filesystem already frozen, return 201
                echo "ERROR: Failed to freeze mountpoint $targetdue due to error -
$errormessage"
                thaw_db
                exit 201
            fi
            echo "INFO: Freezing complete on $target"
        done
    }

    # Iterate over all the mountpoints and unfreeze the filesystem.
    unfreeze_fs() {
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
            # Freeze of the root and boot filesystems is dangerous and pre-script
does not freeze these filesystems.
            # Hence, will skip the root and boot mountpoints during unfreeze as
well.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi
            echo "INFO: Thawing $target"
            error_message=$(sudo fsfreeze -u $target 2>&1)
            # Check if filesystem is already unfrozen (thawed). Return error code
204 if filesystem is already unfrozen.
            if [ $? -ne 0 ]; then
                if [[ "$error_message" == *"$FS_ALREADY_THAWED_ERROR"* ]]; then
                    echo "ERROR: Filesystem ${target} is already in thaw state.
Return Error Code: 205"
```

```
                    exit 205
                fi
                # If the filesystem unfreeze failed due to any reason other than
 the filesystem already unfrozen, return 202
                echo "ERROR: Failed to unfreeze mountpoint $targetdue due to error
 - $errormessage"
                exit 202
            fi
            echo "INFO: Thaw complete on $target"
        done
    }

    snap_db() {
        # Run the flush command only when MySQL DB service is up and running
        sudo systemctl is-active --quiet mysqld.service
        if [ $? -eq 0 ]; then
            echo "INFO: Execute MySQL Flush and Lock command."
            sudo mysql -e 'FLUSH TABLES WITH READ LOCK;'
            # If the MySQL Flush and Lock command did not succeed, return error
 code 201 to indicate pre-script failure
            if [ $? -ne 0 ]; then
                echo "ERROR: MySQL FLUSH TABLES WITH READ LOCK command failed."
                exit 201
            fi
            sync
        else
            echo "INFO: MySQL service is inactive. Skipping execution of MySQL
 Flush and Lock command."
        fi
    }

    thaw_db() {
        # Run the unlock command only when MySQL DB service is up and running
        sudo systemctl is-active --quiet mysqld.service
        if [ $? -eq 0 ]; then
            echo "INFO: Execute MySQL Unlock"
            sudo mysql -e 'UNLOCK TABLES;'
        else
            echo "INFO: MySQL service is inactive. Skipping execution of MySQL
 Unlock command."
        fi
    }

    export -f execute_schedule_auto_thaw
```

```
      export -f execute_post_script
      export -f unfreeze_fs
      export -f thaw_db

      # Debug logging for parameters passed to the SSM document
      echo "INFO: ${OPERATION} starting at $(date) with executionId:
 ${EXECUTION_ID}"

      # Based on the command parameter value execute the function that supports
      # pre-script/post-script operation
      case ${OPERATION} in
          pre-script)
              execute_pre_script
              ;;
          post-script)
              execute_post_script
              execute_disable_auto_thaw
              ;;
          dry-run)
              echo "INFO: dry-run option invoked - taking no action"
              ;;
          *)
              echo "ERROR: Invalid command parameter passed. Please use either pre-
script, post-script, dry-run."
              exit 1 # return failure
              ;;
      esac

      END=$(date +%s)
      # Debug Log for profiling the script time
      echo "INFO: ${OPERATION} completed at $(date). Total runtime: $((${END} -
 ${START})) seconds."
```

PostgreSQL sample document content

```
###===============================================================================###
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
```

```
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.

# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
###===============================================================================###
schemaVersion: '2.2'
description: Amazon Data Lifecycle Manager Pre/Post script for PostgreSQL databases
parameters:
  executionId:
    type: String
    default: None
    description: (Required) Specifies the unique identifier associated with a pre
 and/or post execution
    allowedPattern: ^(None|[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]
{4}-[a-fA-F0-9]{12})$
  command:
  # Data Lifecycle Manager will trigger the pre-script and post-script actions
 during policy execution.
  # 'dry-run' option is intended for validating the document execution without
 triggering any commands
  # on the instance. The following allowedValues will allow Data Lifecycle Manager
 to successfully
  # trigger pre and post script actions.
    type: String
    default: 'dry-run'
    description: (Required) Specifies whether pre-script and/or post-script should
 be executed.
    allowedValues:
    - pre-script
    - post-script
    - dry-run

mainSteps:
- action: aws:runShellScript
  description: Run PostgreSQL Database freeze/thaw commands
  name: run_pre_post_scripts
  precondition:
    StringEquals:
```

```
    - platformType
    - Linux
  inputs:
    runCommand:
    - |
      #!/bin/bash


###===============================================================================###
      ### Error Codes

###===============================================================================###
      # The following Error codes will inform Data Lifecycle Manager of the type of
error
      # and help guide handling of the error.
      # The Error code will also be emitted via AWS Eventbridge events in the
'cause' field.
      # 1 Pre-script failed during execution - 201
      # 2 Post-script failed during execution - 202
      # 3 Auto thaw occurred before post-script was initiated - 203
      # 4 Pre-script initiated while post-script was expected - 204
      # 5 Post-script initiated while pre-script was expected - 205
      # 6 Application not ready for pre or post-script initiation - 206


###===============================================================================###
      ### Global variables

###===============================================================================###
      START=$(date +%s)
      OPERATION={{ command }}
      FS_ALREADY_FROZEN_ERROR='freeze failed: Device or resource busy'
      FS_ALREADY_THAWED_ERROR='unfreeze failed: Invalid argument'
      FS_BUSY_ERROR='mount point is busy'

      # Auto thaw is a fail safe mechanism to automatically unfreeze the application
after the
      # duration specified in the global variable below. Choose the duration based
on your
      # database application's tolerance to freeze.
      export AUTO_THAW_DURATION_SECS="60"

      # Add all pre-script actions to be performed within the function below
      execute_pre_script() {
```

```
        echo "INFO: Start execution of pre-script"
        # Check if filesystem is already frozen. No error code indicates that
filesystem
        # is not currently frozen and that the pre-script can proceed with
freezing the filesystem.
        check_fs_freeze
        # Execute the DB commands to flush the DB in preparation for snapshot
        snap_db
        # Freeze the filesystem. No error code indicates that filesystem was
succefully frozen
        freeze_fs

        echo "INFO: Schedule Auto Thaw to execute in ${AUTO_THAW_DURATION_SECS}
seconds."
        $(nohup bash -c execute_schedule_auto_thaw  >/dev/null 2>&1 &)
    }

    # Add all post-script actions to be performed within the function below
    execute_post_script() {
        echo "INFO: Start execution of post-script"
        # Unfreeze the filesystem. No error code indicates that filesystem was
successfully unfrozen
        unfreeze_fs
    }

    # Execute Auto Thaw to automatically unfreeze the application after the
duration configured
    # in the AUTO_THAW_DURATION_SECS global variable.
    execute_schedule_auto_thaw() {
        sleep ${AUTO_THAW_DURATION_SECS}
        execute_post_script
    }

    # Disable Auto Thaw if it is still enabled
    execute_disable_auto_thaw() {
        echo "INFO: Attempting to disable auto thaw if enabled"
        auto_thaw_pgid=$(pgrep -f execute_schedule_auto_thaw | xargs -i ps -hp {}
-o pgid)
        if [ -n "${auto_thaw_pgid}" ]; then
            echo "INFO: execute_schedule_auto_thaw process found with pgid
${auto_thaw_pgid}"
            sudo pkill -g ${auto_thaw_pgid}
            rc=$?
            if [ ${rc} != 0 ]; then
```

```
                echo "ERROR: Unable to kill execute_schedule_auto_thaw process. 
retval=${rc}"
            else
                echo "INFO: Auto Thaw  has been disabled"
            fi
        fi
    }

    # Iterate over all the mountpoints and check if filesystem is already in 
freeze state.
    # Return error code 204 if any of the mount points are already frozen.
    check_fs_freeze() {
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
            # Freeze of the root and boot filesystems is dangerous and pre-script 
does not freeze these filesystems.
            # Hence, we will skip the root and boot mountpoints while checking if 
filesystem is in freeze state.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi

            error_message=$(sudo mount -o remount,noatime $target 2>&1)
            # Remount will be a no-op without a error message if the filesystem is 
unfrozen.
            # However, if filesystem is already frozen, remount will fail with 
busy error message.
            if [ $? -ne 0 ];then
                # If the filesystem is already in frozen, return error code 204
                if [[ "$error_message" == *"$FS_BUSY_ERROR"* ]];then
                    echo "ERROR: Filesystem ${target} already frozen. Return Error 
Code: 204"
                    exit 204
                fi
                # If the check filesystem freeze failed due to any reason other 
than the filesystem already frozen, return 201
                echo "ERROR: Failed to check_fs_freeze on mountpoint $target due 
to error - $errormessage"
                exit 201
            fi
        done
    }

    # Iterate over all the mountpoints and freeze the filesystem.
    freeze_fs() {
```

```
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
            # Freeze of the root and boot filesystems is dangerous. Hence, skip
filesystem freeze
            # operations for root and boot mountpoints.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi
            echo "INFO: Freezing $target"
            error_message=$(sudo fsfreeze -f $target 2>&1)
            if [ $? -ne 0 ];then
                # If the filesystem is already in frozen, return error code 204
                if [[ "$error_message" == *"$FS_ALREADY_FROZEN_ERROR"* ]]; then
                    echo "ERROR: Filesystem ${target} already frozen. Return Error
Code: 204"
                    exit 204
                fi
                # If the filesystem freeze failed due to any reason other than the
filesystem already frozen, return 201
                echo "ERROR: Failed to freeze mountpoint $targetdue due to error -
$errormessage"
                exit 201
            fi
            echo "INFO: Freezing complete on $target"
        done
    }

    # Iterate over all the mountpoints and unfreeze the filesystem.
    unfreeze_fs() {
        for target in $(lsblk -nlo MOUNTPOINTS)
        do
            # Freeze of the root and boot filesystems is dangerous and pre-script
does not freeze these filesystems.
            # Hence, will skip the root and boot mountpoints during unfreeze as
well.
            if [ $target == '/' ]; then continue; fi
            if [[ "$target" == *"/boot"* ]]; then continue; fi
            echo "INFO: Thawing $target"
            error_message=$(sudo fsfreeze -u $target 2>&1)
            # Check if filesystem is already unfrozen (thawed). Return error code
204 if filesystem is already unfrozen.
            if [ $? -ne 0 ]; then
                if [[ "$error_message" == *"$FS_ALREADY_THAWED_ERROR"* ]]; then
                    echo "ERROR: Filesystem ${target} is already in thaw state.
Return Error Code: 205"
```

```
                    exit 205
                fi
                # If the filesystem unfreeze failed due to any reason other than
 the filesystem already unfrozen, return 202
                echo "ERROR: Failed to unfreeze mountpoint $targetdue due to error
 - $errormessage"
                exit 202
            fi
            echo "INFO: Thaw complete on $target"
        done
    }

    snap_db() {
        # Run the flush command only when PostgreSQL DB service is up and running
        sudo systemctl is-active --quiet postgresql
        if [ $? -eq 0 ]; then
            echo "INFO: Execute Postgres CHECKPOINT"
            # PostgreSQL command to flush the transactions in memory to disk
            sudo -u postgres psql -c 'CHECKPOINT;'
            # If the PostgreSQL Command did not succeed, return error code 201 to
 indicate pre-script failure
            if [ $? -ne 0 ]; then
                echo "ERROR: Postgres CHECKPOINT command failed."
                exit 201
            fi
            sync
        else
            echo "INFO: PostgreSQL service is inactive. Skipping execution of
 CHECKPOINT command."
        fi
    }

    export -f execute_schedule_auto_thaw
    export -f execute_post_script
    export -f unfreeze_fs

    # Debug logging for parameters passed to the SSM document
    echo "INFO: ${OPERATION} starting at $(date) with executionId:
 ${EXECUTION_ID}"

    # Based on the command parameter value execute the function that supports
    # pre-script/post-script operation
    case ${OPERATION} in
        pre-script)
```

```
                execute_pre_script
                ;;
            post-script)
                execute_post_script
                execute_disable_auto_thaw
                ;;
            dry-run)
                echo "INFO: dry-run option invoked - taking no action"
                ;;
            *)
                echo "ERROR: Invalid command parameter passed. Please use either pre-
script, post-script, dry-run."
                exit 1 # return failure
                ;;
        esac

        END=$(date +%s)
        # Debug Log for profiling the script time
        echo "INFO: ${OPERATION} completed at $(date). Total runtime: $((${END} -
 ${START})) seconds."
```

InterSystems IRIS sample document content

```
###=====================================================================================###
# MIT License
#
# Copyright (c) 2024 InterSystems
#
# Permission is hereby granted, free of charge, to any person obtaining a copy
# of this software and associated documentation files (the "Software"), to deal
# in the Software without restriction, including without limitation the rights
# to use, copy, modify, merge, publish, distribute, sublicense, and/or sell
# copies of the Software, and to permit persons to whom the Software is
# furnished to do so, subject to the following conditions:
#
# The above copyright notice and this permission notice shall be included in all
# copies or substantial portions of the Software.
#
# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
# IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY,
# FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE
# AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER
# LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM,
```

```
# OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE
# SOFTWARE.
###===============================================================================###
schemaVersion: '2.2'
description: SSM Document Template for Amazon Data Lifecycle Manager Pre/Post script
 feature for InterSystems IRIS.
parameters:
  executionId:
    type: String
    default: None
    description: Specifies the unique identifier associated with a pre and/or post
 execution
    allowedPattern: ^(None|[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]
{4}-[a-fA-F0-9]{12})$
  command:
    type: String
    # Data Lifecycle Manager will trigger the pre-script and post-script actions.
 You can also use this SSM document with 'dry-run' for manual testing purposes.
    default: 'dry-run'
    description: (Required) Specifies whether pre-script and/or post-script should
 be executed.
    #The following allowedValues will allow Data Lifecycle Manager to successfully
 trigger pre and post script actions.
    allowedValues:
    - pre-script
    - post-script
    - dry-run

mainSteps:
- action: aws:runShellScript
  description: Run InterSystems IRIS Database freeze/thaw commands
  name: run_pre_post_scripts
  precondition:
    StringEquals:
    - platformType
    - Linux
  inputs:
    runCommand:
    - |
      #!/bin/bash

 ###===============================================================================###
      ### Global variables
```

```
###===============================================================================###
     DOCKER_NAME=iris
     LOGDIR=./
     EXIT_CODE=0
     OPERATION={{ command }}
     START=$(date +%s)

     # Check if Docker is installed
     # By default if Docker is present, script assumes that InterSystems IRIS is
running in Docker
     # Leave only the else block DOCKER_EXEC line, if you run InterSystems IRIS
non-containerised (and Docker is present).
     # Script assumes irissys user has OS auth enabled, change the OS user or
supply login/password depending on your configuration.
     if command -v docker &> /dev/null
     then
       DOCKER_EXEC="docker exec $DOCKER_NAME"
     else
       DOCKER_EXEC="sudo -i -u irissys"
     fi


     # Add all pre-script actions to be performed within the function below
     execute_pre_script() {
       echo "INFO: Start execution of pre-script"

       # find all iris running instances
       iris_instances=$($DOCKER_EXEC iris qall 2>/dev/null | tail -n +3 | grep
'^up' | cut -c5-  | awk '{print $1}')
       echo "`date`: Running iris instances $iris_instances"

       # Only for running instances
       for INST in $iris_instances; do

         echo "`date`: Attempting to freeze $INST"

         # Detailed instances specific log
         LOGFILE=$LOGDIR/$INST-pre_post.log

         #check Freeze status before starting
         $DOCKER_EXEC irissession $INST -U '%SYS'
"##Class(Backup.General).IsWDSuspendedExt()"
         freeze_status=$?
```

```
          if [ $freeze_status -eq 5 ]; then
            echo "`date`:   ERROR: $INST IS already FROZEN"
            EXIT_CODE=204
          else
            echo "`date`:   $INST is not frozen"
            # Freeze
            # Docs: https://docs.intersystems.com/irislatest/csp/documatic/
%25CSP.Documatic.cls?LIBRARY=%25SYS&CLASSNAME=Backup.General#ExternalFreeze
            $DOCKER_EXEC irissession $INST -U '%SYS'
 "##Class(Backup.General).ExternalFreeze(\"$LOGFILE\",,,,,,600,,,300)"
            status=$?

            case $status in
              5) echo "`date`:   $INST IS FROZEN"
                ;;
              3) echo "`date`:   $INST FREEZE FAILED"
                EXIT_CODE=201
                ;;
              *) echo "`date`:   ERROR: Unknown status code: $status"
                EXIT_CODE=201
                ;;
            esac
            echo "`date`:   Completed freeze of $INST"
          fi
        done
        echo "`date`: Pre freeze script finished"
      }

      # Add all post-script actions to be performed within the function below
      execute_post_script() {
        echo "INFO: Start execution of post-script"

        # find all iris running instances
        iris_instances=$($DOCKER_EXEC iris qall 2>/dev/null | tail -n +3 | grep
 '^up' | cut -c5-  | awk '{print $1}')
        echo "`date`: Running iris instances $iris_instances"

        # Only for running instances
        for INST in $iris_instances; do

          echo "`date`: Attempting to thaw $INST"

          # Detailed instances specific log
          LOGFILE=$LOGDIR/$INST-pre_post.log
```

```
          #check Freeze status befor starting
          $DOCKER_EXEC irissession $INST -U '%SYS'
 "##Class(Backup.General).IsWDSuspendedExt()"
          freeze_status=$?
          if [ $freeze_status -eq 5 ]; then
            echo "`date`:  $INST is in frozen state"
            # Thaw
            # Docs: https://docs.intersystems.com/irislatest/csp/documatic/
%25CSP.Documatic.cls?LIBRARY=%25SYS&CLASSNAME=Backup.General#ExternalFreeze
            $DOCKER_EXEC irissession $INST -U%SYS
 "##Class(Backup.General).ExternalThaw(\"$LOGFILE\")"
            status=$?

            case $status in
              5) echo "`date`:   $INST IS THAWED"
                  $DOCKER_EXEC irissession $INST -U%SYS
 "##Class(Backup.General).ExternalSetHistory(\"$LOGFILE\")"
                ;;
              3) echo "`date`:   $INST THAW FAILED"
                  EXIT_CODE=202
                ;;
              *) echo "`date`:   ERROR: Unknown status code: $status"
                  EXIT_CODE=202
                ;;
            esac
            echo "`date`:   Completed thaw of $INST"
          else
            echo "`date`:   ERROR: $INST IS already THAWED"
            EXIT_CODE=205
          fi
        done
        echo "`date`: Post thaw script finished"
      }

      # Debug logging for parameters passed to the SSM document
        echo "INFO: ${OPERATION} starting at $(date) with executionId:
 ${EXECUTION_ID}"

      # Based on the command parameter value execute the function that supports
      # pre-script/post-script operation
      case ${OPERATION} in
        pre-script)
          execute_pre_script
```

```
            ;;
          post-script)
            execute_post_script
              ;;
          dry-run)
            echo "INFO: dry-run option invoked - taking no action"
            ;;
          *)
            echo "ERROR: Invalid command parameter passed. Please use either pre-
script, post-script, dry-run."
            # return failure
            EXIT_CODE=1
            ;;
        esac

        END=$(date +%s)
        # Debug Log for profiling the script time
        echo "INFO: ${OPERATION} completed at $(date). Total runtime: $((${END} -
 ${START})) seconds."
        exit $EXIT_CODE
```

For more information, see the [GitHub repository](https://github.com/).

Empty document template

```
###===============================================================================###
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# Permission is hereby granted, free of charge, to any person obtaining a copy of
 this
# software and associated documentation files (the "Software"), to deal in the
 Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.

# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR
 IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
###===============================================================================###
```

```
schemaVersion: '2.2'
description: SSM Document Template for Amazon Data Lifecycle Manager Pre/Post script
 feature
parameters:
  executionId:
    type: String
    default: None
    description: (Required) Specifies the unique identifier associated with a pre
 and/or post execution
    allowedPattern: ^(None|[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]
{4}-[a-fA-F0-9]{12})$
  command:
  # Data Lifecycle Manager will trigger the pre-script and post-script actions
 during policy execution.
  # 'dry-run' option is intended for validating the document execution without
 triggering any commands
  # on the instance. The following allowedValues will allow Data Lifecycle Manager
 to successfully
  # trigger pre and post script actions.
    type: String
    default: 'dry-run'
    description: (Required) Specifies whether pre-script and/or post-script should
 be executed.
    allowedValues:
    - pre-script
    - post-script
    - dry-run

mainSteps:
- action: aws:runShellScript
  description: Run Database freeze/thaw commands
  name: run_pre_post_scripts
  precondition:
    StringEquals:
    - platformType
    - Linux
  inputs:
    runCommand:
    - |
      #!/bin/bash


 ###===============================================================================###
      ### Error Codes
```

```
###===============================================================================###
     # The following Error codes will inform Data Lifecycle Manager of the type of
error
     # and help guide handling of the error.
     # The Error code will also be emitted via AWS Eventbridge events in the
'cause' field.
     # 1 Pre-script failed during execution - 201
     # 2 Post-script failed during execution - 202
     # 3 Auto thaw occurred before post-script was initiated - 203
     # 4 Pre-script initiated while post-script was expected - 204
     # 5 Post-script initiated while pre-script was expected - 205
     # 6 Application not ready for pre or post-script initiation - 206


###===============================================================================###
     ### Global variables

###===============================================================================###
     START=$(date +%s)
     # For testing this script locally, replace the below with OPERATION=$1.
     OPERATION={{ command }}

     # Add all pre-script actions to be performed within the function below
     execute_pre_script() {
         echo "INFO: Start execution of pre-script"
     }

     # Add all post-script actions to be performed within the function below
     execute_post_script() {
         echo "INFO: Start execution of post-script"
     }

     # Debug logging for parameters passed to the SSM document
     echo "INFO: ${OPERATION} starting at $(date) with executionId:
${EXECUTION_ID}"

     # Based on the command parameter value execute the function that supports
     # pre-script/post-script operation
     case ${OPERATION} in
         pre-script)
             execute_pre_script
             ;;
         post-script)
```

```
                execute_post_script
                ;;
            dry-run)
                echo "INFO: dry-run option invoked - taking no action"
                ;;
            *)
                echo "ERROR: Invalid command parameter passed. Please use either pre-
script, post-script, dry-run."
                exit 1 # return failure
                ;;
        esac

        END=$(date +%s)
        # Debug Log for profiling the script time
        echo "INFO: ${OPERATION} completed at $(date). Total runtime: $((${END} -
 ${START})) seconds."
```

Once you have your SSM document content, use one of the following procedures to create the custom SSM document.

Console

To create the SSM command document

1. Open the AWS Systems Manager console at https://console.aws.amazon.com//systems-manager/.
2. In the navigation pane, choose **Documents**, then choose **Create document**, **Command or Session**.
3. For **Name**, enter a descriptive name for the document.
4. For **Target type**, select **/AWS::EC2::Instance**.
5. For **Document type**, select **Command**.
6. In the **Content** field, select **YAML** and then paste the document content.
7. In the **Document tags** section, add a tag with a tag key of `DLMScriptsAccess`, and a tag value of `true`.

> **Important**
>
> The `DLMScriptsAccess:true` tag is required by the **AWSDataLifecycleManagerSSMFullAccess** AWS managed policy used in *Step 3: Prepare Amazon Data Lifecycle Manager IAM role*. The policy uses the `aws:ResourceTag` condition key to restrict access to SSM documents that have this tag.

8. Choose **Create document**.

AWS CLI

To create the SSM command document

Use the create-document command. For `--name`, specify a descriptive name for the document. For `--document-type`, specify `Command`. For `--content`, specify the path to the .yaml file with the SSM document content. For `--tags`, specify `"Key=DLMScriptsAccess,Value=true"`.

```
$ aws ssm create-document \
--content file://path/to/file/documentContent.yaml \
--name "document_name" \
--document-type "Command" \
--document-format YAML \
--tags "Key=DLMScriptsAccess,Value=true"
```

Step 3: Prepare Amazon Data Lifecycle Manager IAM role

> **Note**
>
> This step is needed if:
>
> - You create or update a pre/post script-enabled snapshot policy that uses a custom IAM role.
> - You use the command line to create or update a pre/post script-enabled snapshot policy that uses the default.

If you use the console to create or update a pre/post script-enabled snapshot policy that uses the default role for managing snapshots (**AWSDataLifecycleManagerDefaultRole**), skip this step. In this case, we automatically attach the **AWSDataLifecycleManagerSSMFullAccess** policy to that role.

You must ensure that the IAM role that you use for policy grants Amazon Data Lifecycle Manager permission to perform the SSM actions required to run pre and post scripts on instances targeted by the policy.

Amazon Data Lifecycle Manager provides a managed policy (**AWSDataLifecycleManagerSSMFullAccess**) that includes the required permissions. You can attach this policy to your IAM role for managing snapshots to ensure that it includes the permissions.

Important

The AWSDataLifecycleManagerSSMFullAccess managed policy uses the `aws:ResourceTag` condition key to restrict access to specific SSM documents when using pre and post scripts. To allow Amazon Data Lifecycle Manager to access the SSM documents, you must ensure that your SSM documents are tagged with `DLMScriptsAccess:true`.

Alternatively, you can manually create a custom policy or assign the required permissions directly to the IAM role that you use. You can use the same permissions that are defined in the AWSDataLifecycleManagerSSMFullAccess managed policy, however, the `aws:ResourceTag` condition key is optional. If you decide to not include that condition key, then you do not need to tag your SSM documents with `DLMScriptsAccess:true`.

Use one of the following methods to add the **AWSDataLifecycleManagerSSMFullAccess** policy to your IAM role.

Console

To attach the managed policy to your custom role

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation panel, choose **Roles**.

3. Search for and select your custom role for managing snapshots.
4. On the **Permissions** tab, choose **Add permissions**, **Attach policies**.
5. Search for and select the **AWSDataLifecycleManagerSSMFullAccess** managed policy, and then choose **Add permissions**.

AWS CLI

To attach the managed policy to your custom role

Use the attach-role-policy command. For `---role-name`, specify the name of your custom role. For `--policy-arn`, specify `arn:aws:iam::aws:policy/AWSDataLifecycleManagerSSMFullAccess`.

```
$ aws iam attach-role-policy \
--policy-arn arn:aws:iam::aws:policy/AWSDataLifecycleManagerSSMFullAccess \
--role-name your_role_name
```

Step 4: Create snapshot lifecycle policy

To automate application-consistent snapshots, you must create a snapshot lifecycle policy that targets instances, and configure pre and post scripts for that policy.

Console

To create the snapshot lifecycle policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**, and then choose **Create lifecycle policy**.
3. On the **Select policy type** screen, choose **EBS snapshot policy** and then choose **Next**.
4. In the **Target resources** section, do the following:
    a. For **Target resource types**, choose `Instance`.
    b. For **Target resource tags**, specify the resource tags that identify the instances to back up. Only resources that have the specified tags will be backed up.

5. For **IAM role**, either choose **AWSDataLifecycleManagerDefaultRole** (the default role for managing snapshots), or choose a custom role that you created and prepared for pre and post scripts.
6. Configure the schedules and additional options as needed. We recommend that you schedule snapshot creation times for time periods that match your workload, such as during maintenance windows.

   For SAP HANA, we recommend that you enable fast snapshot restore.

   > **Note**
   >
   > If you enable a schedule for VSS Backups, you can't enable **Exclude specific data volumes** or **Copy tags from source**.

7. In the **Pre and post scripts** section, select **Enable pre and post scripts**, and then do the following, depending on your workload:

   - To create application-consistent snapshots of your Windows applications, select **VSS Backup**.
   - To create application-consistent snapshots of your SAP HANA workloads, select **SAP HANA**.
   - To create application-consistent snapshots of all other databases and workloads, including your self-managed MySQL, PostgreSQL, or InterSystems IRIS databases, using a custom SSM document, select **Custom SSM document**.
     1. For **Automate option**, choose **Pre and post scripts**.
     2. For **SSM document**, select the SSM document that you prepared.
8. Depending on the option you selected, configure the following additional options:

   - **Script timeout** — (*Custom SSM document only*) The timeout period after which Amazon Data Lifecycle Manager fails the script run attempt if it has not completed. If a script does not complete within its timeout period, Amazon Data Lifecycle Manager fails the attempt. The timeout period applies to the pre and post scripts individually. The minimum and default timeout period is 10 seconds. And the maximum timeout period is 120 seconds.
   - **Retry failed scripts** — Select this option to retry scripts that do not complete within their timeout period. If the pre script fails, Amazon Data Lifecycle Manager retries entire snapshot creation process, including running the pre and post scripts. If the post script

fails, Amazon Data Lifecycle Manager retries the post script only; in this case, the pre script will have completed and the snapshot might have been created.

- **Default to crash-consistent snapshots** — Select this option to default to crash-consistent snapshots if the pre script fails to run. This is the default snapshot creation behavior for Amazon Data Lifecycle Manager if pre and post scripts is not enabled. If you enabled retries, Amazon Data Lifecycle Manager will default to crash-consistent snapshots only after all retry attempts have been exhausted. If the pre script fails and you do not default to crash-consistent snapshots, Amazon Data Lifecycle Manager will not create snapshots for the instance during that schedule run.

> **Note**
>
> If you are creating snapshots for SAP HANA, then you might want to disabled this option. Crash-consistent snapshots of SAP HANA workloads can't restored in the same manner.

9. Choose **Create default policy**.

> **Note**
>
> If you get the `Role with name AWSDataLifecycleManagerDefaultRole already exists` error, see [Troubleshoot Amazon Data Lifecycle Manager issues](#) for more information.

AWS CLI

To create the snapshot lifecycle policy

Use the [create-lifecycle-policy](#) command, and include the `Scripts` parameters in `CreateRule`. For more information about the parameters, see the *[Amazon Data Lifecycle Manager API Reference](#)*.

```
$ aws dlm create-lifecycle-policy \
--description "policy_description" \
--state ENABLED \
--execution-role-arn iam_role_arn \
--policy-details file://policyDetails.json
```

Where `policyDetails.json` includes one of the following, depending on your use case:

- **VSS Backup**

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key": "tag_key",
        "Value": "tag_value"
    }],
    "Schedules": [{
        "Name": "schedule_name",
        "CreateRule": {
            "CronExpression": "cron_for_creation_frequency",
            "Scripts": [{
                "ExecutionHandler":"AWS_VSS_BACKUP",
                "ExecuteOperationOnScriptFailure":true|false,
                "MaximumRetryCount":retries (0-3)
            }]
        },
        "RetainRule": {
            "Count": retention_count
        }
    }]
}
```

- **SAP HANA backups**

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key": "tag_key",
        "Value": "tag_value"
    }],
    "Schedules": [{
        "Name": "schedule_name",
        "CreateRule": {
```

```
            "CronExpression": "cron_for_creation_frequency",
            "Scripts": [{
                "Stages": ["PRE","POST"],
                "ExecutionHandlerService":"AWS_SYSTEMS_MANAGER",
                "ExecutionHandler":"AWSSystemsManagerSAP-
CreateDLMSnapshotForSAPHANA",
                "ExecuteOperationOnScriptFailure":true|false,
                "ExecutionTimeout":timeout_in_seconds (10-120),
                "MaximumRetryCount":retries (0-3)
            }]
        },
        "RetainRule": {
            "Count": retention_count
        }
    }]
}
```

- **Custom SSM document**

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key": "tag_key",
        "Value": "tag_value"
    }],
    "Schedules": [{
        "Name": "schedule_name",
        "CreateRule": {
            "CronExpression": "cron_for_creation_frequency",
            "Scripts": [{
                "Stages": ["PRE","POST"],
                "ExecutionHandlerService":"AWS_SYSTEMS_MANAGER",
                "ExecutionHandler":"ssm_document_name|arn",
                "ExecuteOperationOnScriptFailure":true|false,
                "ExecutionTimeout":timeout_in_seconds (10-120),
                "MaximumRetryCount":retries (0-3)
            }]
        },
        "RetainRule": {
            "Count": retention_count
        }
```

```
    }]
}
```

Considerations for VSS Backups with Amazon Data Lifecycle Manager

With Amazon Data Lifecycle Manager, you can back up and restore VSS (Volume Shadow Copy Service)-enabled Windows applications running on Amazon EC2 instances. If the application has a VSS writer registered with Windows VSS, then Amazon Data Lifecycle Manager creates a snapshot that will be application-consistent for that application.

> **Note**
>
> Amazon Data Lifecycle Manager currently supports application-consistent snapshots of resources running on Amazon EC2 only, specifically for backup scenarios where application data can be restored by replacing an existing instance with a new instance created from the backup. Not all instance types or applications are supported for VSS backups. For more information, see Application-consistent Windows VSS snapshots in the *Amazon EC2 User Guide*.

Unsupported instance types

The following Amazon EC2 instance types are not supported for VSS backups. If your policy targets one of these instance types, Amazon Data Lifecycle Manager might still create VSS backups, but the snapshots might not be tagged with the required system tags. Without these tags, the snapshots will not be managed by Amazon Data Lifecycle Manager after creation. You might need to manually delete these snapshots.

- T3: `t3.nano` | `t3.micro`
- T3a: `t3a.nano` | `t3a.micro`
- T2: `t2.nano` | `t2.micro`

Shared responsibility for application-consistent snapshots

You must ensure that:

- The SSM Agent is installed, up-to-date, and running on your target instances

- Systems Manager has permissions to perform the required actions on the target instances
- Amazon Data Lifecycle Manager has permissions to perform the Systems Manager actions required to run pre and post scripts on the target instances.
- For custom workloads, such as self-managed MySQL, PostgreSQL, or InterSystems IRIS databases, the SSM document that you use includes the correct and required actions for freezing, flushing, and thawing I/O for your database configuration.
- Snapshot creation times align with your workload schedule. For example, try to schedule snapshot creation during scheduled maintenance windows.

Amazon Data Lifecycle Manager ensures that:

- Snapshot creation is initiated within 60 minutes of the scheduled snapshot creation time.
- Pre scripts run before the snapshot creation is initiated.
- Post scripts run after the pre script succeeds and the snapshot creation has been initiated. Amazon Data Lifecycle Manager runs the post script only if the pre script succeeds. If the pre script fails, Amazon Data Lifecycle Manager will not run the post script.
- Snapshots are tagged with the appropriate tags on creation.
- CloudWatch metrics and events are emitted when scripts are initiated, and when they fail or succeed.

Other use cases for Data Lifecycle Manager pre and post scripts

In addition to using pre and post scripts for automating application-consistent snapshots, you can use pre and post scripts together, or individually, to automate other administrative tasks before or after snapshot creation. For example:

- Using a pre script to apply patches before creating snapshots. This can help you create snapshots after applying your regular weekly or monthly software updates.

> **Note**
>
> If you choose to run a pre script only, **Default to crash-consistent snapshots** is enabled by default.

- Using a post script to apply patches after creating snapshots. This can help you create snapshots before applying your regular weekly or monthly software updates.

Getting started for other use cases

This section explains the steps you need perform when using pre and/or post scripts for **uses cases other than application-consistent snapshots**.

Step 1: Prepare target instances

To prepare your target instances for pre and/or post scripts

1. Install the SSM Agent on your target instances, if it is not already installed. If SSM Agent is already installed on your target instances, skip this step.
   - (Linux instances) Manually installing SSM Agent on EC2 instances for Linux
   - (Windows instances) Working with SSM Agent on EC2 instances for Windows Server
2. Ensure that the SSM Agent is running. For more information, see Checking SSM Agent status and starting the agent.
3. Set up Systems Manager for Amazon EC2 instances. For more information, see Setting up Systems Manager for Amazon EC2 instances in the *AWS Systems Manager User Guide*.

Step 2: Prepare SSM document

You must create an SSM command document that includes the pre and/or post scripts with the commands you want to run.

You can create an SSM document using the empty SSM document template below and add your pre and post script commands in the appropriate document sections.

Note the following:

- It is your responsibility to ensure that the SSM document performs the correct and required actions for your workload.
- The SSM document must include required fields for `allowedValues`, including `pre-script`, `post-script`, and `dry-run`. Amazon Data Lifecycle Manager will execute commands on your instance based on the contents of those sections. If your SSM document does not have those sections, then Amazon Data Lifecycle Manager will treat it as a failed execution.

```
###===============================================================================###
# Copyright Amazon.com, Inc. or its affiliates. All Rights Reserved.

# Permission is hereby granted, free of charge, to any person obtaining a copy of this
# software and associated documentation files (the "Software"), to deal in the Software
# without restriction, including without limitation the rights to use, copy, modify,
# merge, publish, distribute, sublicense, and/or sell copies of the Software, and to
# permit persons to whom the Software is furnished to do so.

# THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED,
# INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A
# PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT
# HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION
# OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE
# SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.
###===============================================================================###
schemaVersion: '2.2'
description: SSM Document Template for Amazon Data Lifecycle Manager Pre/Post script
 feature
parameters:
  executionId:
    type: String
    default: None
    description: (Required) Specifies the unique identifier associated with a pre and/
or post execution
    allowedPattern: ^(None|[a-fA-F0-9]{8}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-[a-fA-F0-9]{4}-
[a-fA-F0-9]{12})$
  command:
  # Data Lifecycle Manager will trigger the pre-script and post-script actions during
 policy execution.
  # 'dry-run' option is intended for validating the document execution without
 triggering any commands
  # on the instance. The following allowedValues will allow Data Lifecycle Manager to
 successfully
  # trigger pre and post script actions.
    type: String
    default: 'dry-run'
    description: (Required) Specifies whether pre-script and/or post-script should be
 executed.
    allowedValues:
    - pre-script
    - post-script
    - dry-run
```

```
mainSteps:
- action: aws:runShellScript
  description: Run Database freeze/thaw commands
  name: run_pre_post_scripts
  precondition:
    StringEquals:
    - platformType
    - Linux
  inputs:
    runCommand:
    - |
      #!/bin/bash


 ###===============================================================================###
      ### Error Codes

 ###===============================================================================###
      # The following Error codes will inform Data Lifecycle Manager of the type of
 error
      # and help guide handling of the error.
      # The Error code will also be emitted via AWS Eventbridge events in the 'cause'
 field.
      # 1 Pre-script failed during execution - 201
      # 2 Post-script failed during execution - 202
      # 3 Auto thaw occurred before post-script was initiated - 203
      # 4 Pre-script initiated while post-script was expected - 204
      # 5 Post-script initiated while pre-script was expected - 205
      # 6 Application not ready for pre or post-script initiation - 206


 ###===============================================================================###
      ### Global variables

 ###===============================================================================###
      START=$(date +%s)
      # For testing this script locally, replace the below with OPERATION=$1.
      OPERATION={{ command }}

      # Add all pre-script actions to be performed within the function below
      execute_pre_script() {
          echo "INFO: Start execution of pre-script"
      }
```

```
    # Add all post-script actions to be performed within the function below
    execute_post_script() {
        echo "INFO: Start execution of post-script"
    }

    # Debug logging for parameters passed to the SSM document
    echo "INFO: ${OPERATION} starting at $(date) with executionId: ${EXECUTION_ID}"

    # Based on the command parameter value execute the function that supports
    # pre-script/post-script operation
    case ${OPERATION} in
        pre-script)
            execute_pre_script
            ;;
        post-script)
            execute_post_script
            ;;
        dry-run)
            echo "INFO: dry-run option invoked - taking no action"
            ;;
        *)
            echo "ERROR: Invalid command parameter passed. Please use either pre-
script, post-script, dry-run."
            exit 1 # return failure
            ;;
    esac

    END=$(date +%s)
    # Debug Log for profiling the script time
    echo "INFO: ${OPERATION} completed at $(date). Total runtime: $((${END} -
 ${START})) seconds."
```

Step 3: Prepare Amazon Data Lifecycle Manager IAM role

Note

This step is needed if:

- You create or update a pre/post script-enabled snapshot policy that uses a custom IAM role.

- You use the command line to create or update a pre/post script-enabled snapshot policy that uses the default.

If you use the console to create or update a pre/post script-enabled snapshot policy that uses the default role for managing snapshots (**AWSDataLifecycleManagerDefaultRole**), skip this step. In this case, we automatically attach the **AWSDataLifecycleManagerSSMFullAccess** policy to that role.

You must ensure that that IAM role that you use for the policy grants Amazon Data Lifecycle Manager permission to perform the SSM actions required to run pre and post scripts on instances targeted by the policy.

Amazon Data Lifecycle Manager provides a managed policy (**AWSDataLifecycleManagerSSMFullAccess**) that includes the required permissions. You can attach this policy to your IAM role for managing snapshots to ensure that it includes the permissions.

Important

The AWSDataLifecycleManagerSSMFullAccess managed policy uses the `aws:ResourceTag` condition key to restrict access to specific SSM documents when using pre and post scripts. To allow Amazon Data Lifecycle Manager to access the SSM documents, you must ensure that your SSM documents are tagged with `DLMScriptsAccess:true`.

Alternatively, you can manually create a custom policy or assign the required permissions directly to the IAM role that you use. You can use the same permissions that are defined in the AWSDataLifecycleManagerSSMFullAccess managed policy, however, the `aws:ResourceTag` condition key is optional. If you decide to not use that condition key, then you do not need to tag your SSM documents with `DLMScriptsAccess:true`.

Use one of the following methods to add the **AWSDataLifecycleManagerSSMFullAccess** policy to your IAM role.

Console

To attach the managed policy to your custom role

1. Open the IAM console at https://console.aws.amazon.com/iam/.
2. In the navigation panel, choose **Roles**.
3. Search for and select your custom role for managing snapshots.
4. On the **Permissions** tab, choose **Add permissions**, **Attach policies**.
5. Search for and select the **AWSDataLifecycleManagerSSMFullAccess** managed policy, and then choose **Add permissions**.

AWS CLI

To attach the managed policy to your custom role

Use the attach-role-policy command. For `---role-name`, specify the name of your custom role. For `--policy-arn`, specify `arn:aws:iam::aws:policy/AWSDataLifecycleManagerSSMFullAccess`.

```
$ aws iam attach-role-policy \
--policy-arn arn:aws:iam::aws:policy/AWSDataLifecycleManagerSSMFullAccess \
--role-name your_role_name
```

Create snapshot lifecycle policy

Console

To create the snapshot lifecycle policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**, and then choose **Create lifecycle policy**.
3. On the **Select policy type** screen, choose **EBS snapshot policy** and then choose **Next**.
4. In the **Target resources** section, do the following:

    a. For **Target resource types**, choose `Instance`.

   b. For **Target resource tags**, specify the resource tags that identify the instances to back up. Only resources that have the specified tags will be backed up.

5. For **IAM role**, either choose **AWSDataLifecycleManagerDefaultRole** (the default role for managing snapshots), or choose a custom role that you created and prepared for pre and post scripts.
6. Configure the schedules and additional options as needed. We recommend that you schedule snapshot creation times for time periods that match your workload, such as during maintenance windows.
7. In the **Pre and post scripts** section, select **Enable pre and post scripts** and then do the following:

   a. Select **Custom SSM document**.

   b. For **Automate option**, choose the option that matches the scripts you want to run.

   c. For **SSM document**, select the SSM document that you prepared.
8. Configure the following additional options if needed:

   - **Script timeout** — The timeout period after which Amazon Data Lifecycle Manager fails the script run attempt if it has not completed. If a script does not complete within its timeout period, Amazon Data Lifecycle Manager fails the attempt. The timeout period applies to the pre and post scripts individually. The minimum and default timeout period is 10 seconds. And the maximum timeout period is 120 seconds.
   - **Retry failed scripts** — Select this option to retry scripts that do not complete within their timeout period. If the pre script fails, Amazon Data Lifecycle Manager retries entire snapshot creation process, including running the pre and post scripts. If the post script fails, Amazon Data Lifecycle Manager retries the post script only; in this case, the pre script will have completed and the snapshot might have been created.
   - **Default to crash-consistent snapshots** — Select this option to default to crash-consistent snapshots if the pre script fails to run. This is the default snapshot creation behavior for Amazon Data Lifecycle Manager if pre and post scripts is not enabled. If you enabled retries, Amazon Data Lifecycle Manager will default to crash-consistent snapshots only after all retry attempts have been exhausted. If the pre script fails and you do not default to crash-consistent snapshots, Amazon Data Lifecycle Manager will not create snapshots for the instance during that schedule run.
9. Choose **Create default policy**.

Note

If you get the `Role with name AWSDataLifecycleManagerDefaultRole already exists` error, see [Troubleshoot Amazon Data Lifecycle Manager issues](#) for more information.

AWS CLI

To create the snapshot lifecycle policy

Use the [create-lifecycle-policy](#) command, and include the `Scripts` parameters in `CreateRule`. For more information about the parameters, see the *Amazon Data Lifecycle Manager API Reference*.

```
$ aws dlm create-lifecycle-policy \
--description "policy_description" \
--state ENABLED \
--execution-role-arn iam_role_arn \
--policy-details file://policyDetails.json
```

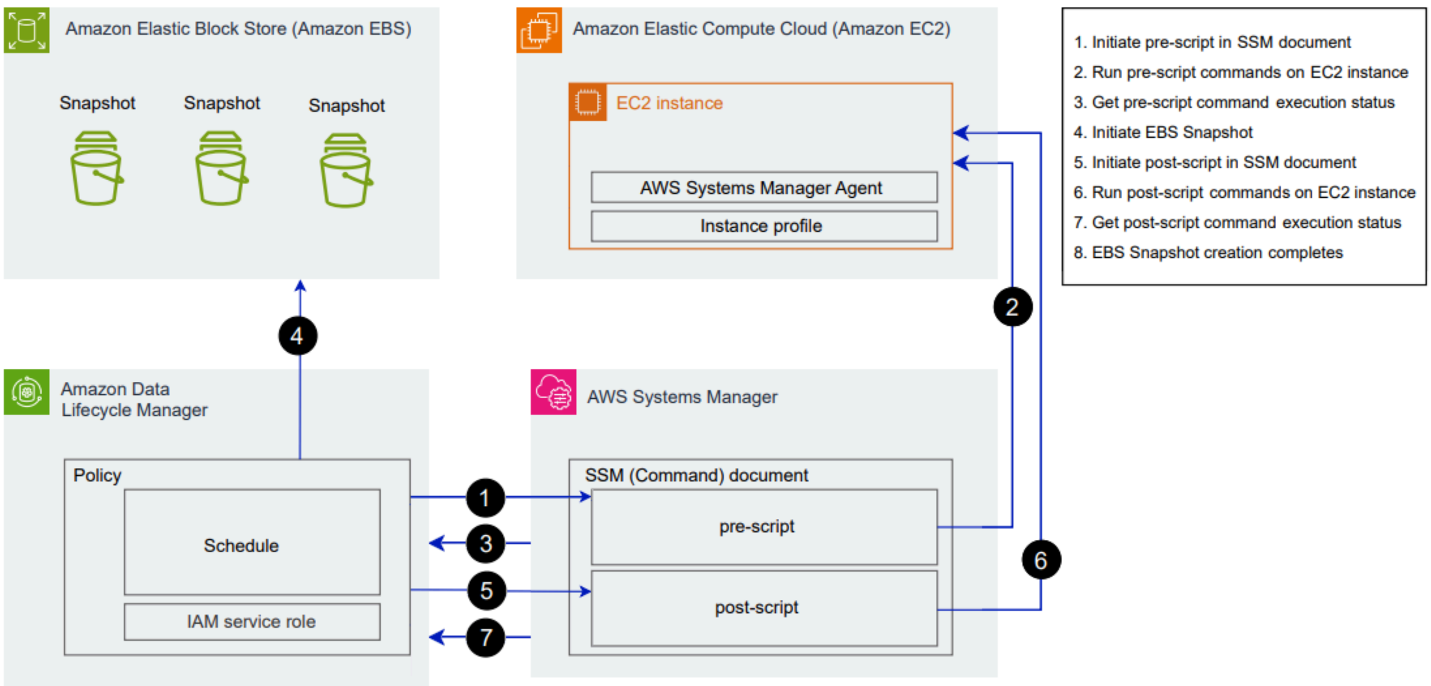
Where `policyDetails.json` includes the following.

```
{
    "PolicyType": "EBS_SNAPSHOT_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key": "tag_key",
        "Value": "tag_value"
    }],
    "Schedules": [{
        "Name": "schedule_name",
        "CreateRule": {
            "CronExpression": "cron_for_creation_frequency",
            "Scripts": [{
                "Stages": ["PRE" | "POST" | "PRE","POST"],
                "ExecutionHandlerService":"AWS_SYSTEMS_MANAGER",
                "ExecutionHandler":"ssm_document_name|arn",
```

```
                    "ExecuteOperationOnScriptFailure":true|false,
                    "ExecutionTimeout":timeout_in_seconds (10-120),
                    "MaximumRetryCount":retries (0-3)
                }]
            },
            "RetainRule": {
                "Count": retention_count
            }
        }]
}
```

How Amazon Data Lifecycle Manager pre and post scripts work

The following image shows the process flow for pre and post scripts when using custom SSM documents. This does not apply to VSS Backups.



At the scheduled snapshot creation time, the following actions and cross-service interactions occur.

1. Amazon Data Lifecycle Manager initiates the pre script action by calling the SSM document and passing the `pre-script` parameter.

> **Note**
>
> Steps 1 to 3 occur only if you run pre scripts. If you run post scripts only, steps 1 to 3 are skipped.

2. Systems Manager sends pre script commands to the SSM Agent running on the target instances. The SSM Agent runs the commands on the instance, and sends status information back to Systems Manager.

   For example, if the SSM document is used to create application-consistent snapshots, the pre script might freeze and flush I/O to ensure that all buffered data is written to the volume before the snapshot is taken.

3. Systems Manager sends pre script command status updates to Amazon Data Lifecycle Manager. If the pre script fails, Amazon Data Lifecycle Manager takes one of the following actions, depending on how you configure the pre and post script options:

Retries	**Default to crash-consistent snapshots**	**Action**
Enabled with retries remaining	Enabled	Retry script until it succeeds or retries are exhausted
Exhausted without successful completion	Enabled	Create crash-consistent snapshots, and do not run post script.
Enabled with retries remaining	Disabled	Retry script until it succeeds or retries are exhausted
Exhausted without successful completion	Disabled	Skip snapshot creation for the target instance, and do not run post script.

Retries	Default to crash-consistent snapshots	Action
Disabled	Enabled	Create crash-consistent snapshots, and do not run post script.
Disabled	Disabled	Skip snapshot creation for the target instance, and do not run post script.

4. Amazon Data Lifecycle Manager initiates snapshot creation.
5. Amazon Data Lifecycle Manager initiates the post script action by calling the SSM document and passing the `post-script` parameter.

> **Note**
>
> Steps 5 to 7 occur only if you run pre scripts. If you run post scripts only, steps 1 to 3 are skipped.

6. Systems Manager sends post script commands to the SSM Agent running on the target instances. The SSM Agent runs the commands on the instance, and sends status information back to Systems Manager.

   For example, if the SSM document enables application-consistent snapshots, this post script might thaw I/O to ensure that your databases resume normal I/O operations after the snapshot has been taken.
7. If you run a post script and Systems Manager indicates that it completed successfully, the process completes.

   If the post script fails, Amazon Data Lifecycle Manager takes one of the following actions, depending on how you configure the pre and post script options:

Retries	Action
Enabled with retries remaining	Retry post script until it succeeds or retries are exhausted
Exhausted without success	Skip post script

Retries	Action
Disabled	Skip post script

Keep in mind that if the post script fails, the pre script (if enabled) will have completed successfully, and the snapshots might have been created. You might need to take further action on the instance to ensure that it is operating as expected. For example if the pre script paused and flushed I/O, but the post script failed to thaw I/O, you might need to configure your database to auto-thaw I/O or you need to manually thaw I/O.

8. The snapshot creation process might complete after the post script completes. The time taken to complete the snapshot depends on the snapshot size.

Identify snapshots created with Data Lifecycle Manager pre and post scripts

Amazon Data Lifecycle Manager automatically assigns the following system tags to snapshots created with pre and post scripts.

- Key: `aws:dlm:pre-script`; Value: `SUCCESS`|`FAILED`

  A tag value of `SUCCESS` indicates that the pre script executed successfully. A tag value of `FAILED` indicates that the pre script did not execute successfully.

- Key: `aws:dlm:post-script`; Value: `SUCCESS`|`FAILED`

  A tag value of `SUCCESS` indicates that the post script executed successfully. A tag value of `FAILED` indicates that the post script did not execute successfully.

For custom SSM documents and SAP HANA backups, you can infer successful application-consistent snapshot creation if the snapshot is tagged with both `aws:dlm:pre-script:SUCCESS` and `aws:dlm:post-script:SUCCESS`.

Additionally, application-consistent snapshots created using VSS backup are automatically tagged with:

- Key: `AppConsistent tag`; Value: `true`|`false`

A tag value of `true` indicates that the VSS backup succeeded and that the snapshots are application-consistent. A tag value of `false` indicates that the VSS backup did not succeed and that the snapshots are not application-consistent.

Monitor Amazon Data Lifecycle Manager pre and post scripts

Amazon CloudWatch metrics

Amazon Data Lifecycle Manager publishes the following CloudWatch metrics when pre and post scripts fail and succeed and when VSS backups fail and succeed.

- `PreScriptStarted`
- `PreScriptCompleted`
- `PreScriptFailed`
- `PostScriptStarted`
- `PostScriptCompleted`
- `PostScriptFailed`
- `VSSBackupStarted`
- `VSSBackupCompleted`
- `VSSBackupFailed`

For more information, see [Monitor Data Lifecycle Manager policies using CloudWatch](#).

Amazon EventBridge

Amazon Data Lifecycle Manager emits the following Amazon EventBridge event when a pre or post script is initiated, succeeds, or fails

- `DLM Pre Post Script Notification`

For more information, see [Monitor Data Lifecycle Manager policies using EventBridge](#).

Create Amazon Data Lifecycle Manager custom policy for EBS-backed AMIs

The following procedure shows you how to use Amazon Data Lifecycle Manager to automate EBS-backed AMI lifecycles.

Topics

Create an AMI lifecycle policy

Use one of the following procedures to create an AMI lifecycle policy.

Console

To create an AMI policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**, and then choose **Create lifecycle policy**.
3. On the **Select policy type** screen, choose **EBS-backed AMI policy**, and then choose **Next**.
4. In the **Target resources** section, for **Target resource tags**, choose the resource tags that identify the volumes or instances to back up. The policy backs up only the resources that have the specified tag key and value pairs.
5. For **Description**, enter a brief description for the policy.
6. For **IAM role**, choose the IAM role that has permissions to manage AMIs and snapshot and to describe instances. To use the default role provided by Amazon Data Lifecycle Manager, choose **Default role**. Alternatively, to use a custom IAM role that you previously created, choose **Choose another role**, and then select the role to use.
7. For **Policy tags**, add the tags to apply to the lifecycle policy. You can use these tags to identify and categorize your policies.
8. For **Policy status after creation**, choose **Enable policy** to start running the policy at the next scheduled time, or **Disable policy** to prevent the policy from running. If you do not

enable the policy now, it will not start creating AMIs until you manually enable it after creation.

9. In the **Instance reboot** section, indicate whether instances should be rebooted before AMI creation. To prevent the targeted instances from being rebooted, choose **No**. Choosing **No** could cause data consistency issues. To reboot instances before AMI creation, choose **Yes**. Choosing this ensures data consistency, but could result in multiple targeted instances rebooting simultaneously.
10. Choose **Next**.
11. On the **Configure schedule** screen, configure the policy schedules. A policy can have up to four schedules. Schedule 1 is mandatory. Schedules 2, 3, and 4 are optional. For each policy schedule that you add, do the following:

    a. In the **Schedule details** section do the following:

        i. For **Schedule name**, specify a descriptive name for the schedule.

        ii. For **Frequency** and the related fields, configure the interval between policy runs.

            You can configure policy runs on a daily, weekly, monthly, or yearly schedule. Alternatively, choose **Custom cron expression** to specify an interval of up to one year. For more information, see Cron and rate expressions in the *Amazon EventBridge User Guide*.

        iii. For **Starting at**, specify the time to start the policy runs. The first policy run starts within an hour after the time that you schedule. You must enter the time in the `hh:mm` UTC format.

        iv. For **Retention type**, specify the retention policy for AMIs created by the schedule.

            You can retain AMIs based on either their total count or their age.

            For count-based retention, the range is `1` to `1000`. After the maximum count is reached, the oldest AMI is deregistered when a new one is created.

            For age-based retention, the range is `1` day to `100` years. After the retention period of each AMI expires, it is deregistered.

            > **Note**
            >
            > All schedules must have the same retention type. You can specify the retention type for Schedule 1 only. Schedules 2, 3, and 4 inherit the

retention type from Schedule 1. Each schedule can have its own retention count or period.

b. Configure tagging for AMIs.

   In the **Tagging** section, do the following:

   i. To copy all of the user-defined tags from the source instance to the AMIs created by the schedule, select **Copy tags from source**.

   ii. By default, AMIs created by the schedule are automatically tagged with the ID of the source instance. To prevent this automatic tagging from happening, for **Variable tags**, remove the `instance-id:$(instance-id)` tile.

   iii. To specify additional tags to assign to AMIs created by this schedule, choose **Add tags**.

c. Configure AMI deprecation.

   To deprecate AMIs when they should no longer be used, in the **AMI deprecation** section, select **Enable AMI deprecation for this schedule** and then specify the AMI deprecation rule. The AMI deprecation rule specifies when AMIs are to be deprecated.

   If the schedule uses count-based AMI retention, you must specify the number of oldest AMIs to deprecate. The deprecation count must be less than or equal to the schedule's AMI retention count, and it can't be greater than 1000. For example, if the schedule is configured to retain a maximum of 5 AMIs, then you can configure the scheduled to deprecate up to old 5 oldest AMIs.

   If the schedule uses age-based AMI retention, you must specify the period after which AMIs are to be deprecated. The deprecation count must be less than or equal to the schedule's AMI retention period, and it can't be greater than 10 years (120 months, 520 weeks, or 3650 days). For example, if the schedule is configured to retain AMIs for 10 days, then you can configure the scheduled to deprecate AMIs after periods up to 10 days after creation.

d. Configure cross-Region copying.

   To copy AMIs created by the schedule to different Regions, in the **Cross-Region copy** section, select **Enable cross-Region copy**. You can copy AMIs to up to three additional Regions in your account. You must specify a separate cross-Region copy rule for each destination Region.

For each destination Region, you can specify the following:

- A retention policy for the AMI copy. When the retention period expires, the copy in the destination Region is automatically deregistered.
- Encryption status for the AMI copy. If the source AMI is encrypted, or if encryption by default is enabled, the copied AMIs are always encrypted. If the source AMI is unencrypted and encryption by default is disabled, you can optionally enable encryption. If you do not specify a KMS key, the AMIs are encrypted using the default KMS key for EBS encryption in each destination Region. If you specify a KMS key for the destination Region, then the selected IAM role must have access to the KMS key.
- A deprecation rule for the AMI copy. When the deprecation period expires, the AMI copy is automatically deprecated. The deprecation period must be less than or equal to the copy retention period, and it can't be greater than 10 years.
- Whether to copy all tags or no tags from the source AMI.

> **Note**
>
> Do not exceed the number of concurrent AMI copies per Region.

e. To add additional schedules, choose **Add another schedule**, which is located at the top of the screen. For each additional schedule, complete the fields as described previously in this topic.

f. After you have added the required schedules, choose **Review policy**.

12. Review the policy summary, and then choose **Create policy**.

> **Note**
>
> If you get the `Role with name AWSDataLifecycleManagerDefaultRoleForAMIManagement already exists` error, see Troubleshoot Amazon Data Lifecycle Manager issues for more information.

Command line

Use the create-lifecycle-policy command to create an AMI lifecycle policy. For PolicyType, specify IMAGE_MANAGEMENT.

> **Note**
>
> To simplify the syntax, the following examples use a JSON file, policyDetails.json, that includes the policy details.

Example 1: Age-based retention and AMI deprecation

This example creates an AMI lifecycle policy that creates AMIs of all instances that have a tag key of purpose with a value of production without rebooting the targeted instances. The policy includes one schedule that creates an AMI every day at 01:00 UTC. The policy retains AMIs for 2 days and deprecates them after 1 day. It also copies the tags from the source instance to the AMIs that it creates.

```
aws dlm create-lifecycle-policy \
    --description "My AMI policy" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRoleForAMIManagement \
    --policy-details file://policyDetails.json
```

The following is an example of the policyDetails.json file.

```
{
    "PolicyType": "IMAGE_MANAGEMENT",
    "ResourceTypes": [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key": "purpose",
        "Value": "production"
    }],
    "Schedules": [{
            "Name": "DailyAMIs",
            "TagsToAdd": [{
                "Key": "type",
```

```
                "Value": "myDailyAMI"
            }],
            "CreateRule": {
                "Interval": 24,
                "IntervalUnit": "HOURS",
                "Times": [
                    "01:00"
                ]
            },
            RetainRule":{
                "Interval" : 2,
                "IntervalUnit" : "DAYS"
            },
            DeprecateRule": {
                "Interval" : 1,
                "IntervalUnit" : "DAYS"
            },
            "CopyTags": true
        }
    ],
    "Parameters" : {
        "NoReboot":true
    }
}
```

If the request succeeds, the command returns the ID of the newly created policy. The following is example output.

```
{
   "PolicyId": "policy-9876543210abcdef0"
}
```

Example 2: Count-based retention and AMI deprecation with cross-Region copy

This example creates an AMI lifecycle policy that creates AMIs of all instances that have a tag key of `purpose` with a value of `production` and reboots the target instances. The policy includes one schedule that creates an AMI every 6 hours starting at `17:30` UTC. The policy retains 3 AMIs and automatically deprecates the 2 oldest AMIs. It also has a cross-Region copy rule that copies AMIs to `us-east-1`, retains 2 AMI copies, and automatically deprecates the oldest AMI.

```
aws dlm create-lifecycle-policy \
```

```
    --description "My AMI policy" \
    --state ENABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRoleForAMIManagement \
    --policy-details file://policyDetails.json
```

The following is an example of the `policyDetails.json` file.

```
{
    "PolicyType": "IMAGE_MANAGEMENT",
    "ResourceTypes" : [
        "INSTANCE"
    ],
    "TargetTags": [{
        "Key":"purpose",
        "Value":"production"
    }],
    "Parameters" : {
          "NoReboot": true
    },
    "Schedules" : [{
        "Name" : "Schedule1",
        "CopyTags": true,
        "CreateRule" : {
            "Interval": 6,
            "IntervalUnit": "HOURS",
            "Times" : ["17:30"]
        },
        "RetainRule":{
            "Count" : 3
        },
        "DeprecateRule":{
            "Count" : 2
        },
        "CrossRegionCopyRules": [{
            "TargetRegion": "us-east-1",
            "Encrypted": true,
            "RetainRule":{
                "IntervalUnit": "DAYS",
                "Interval": 2
            },
            "DeprecateRule":{
                "IntervalUnit": "DAYS",
```

```
                "Interval": 1
            },
            "CopyTags": true
        }]
    }]
}
```

Considerations for AMI lifecycle policies

The following **general considerations** apply to creating AMI lifecycle policies:

- AMI lifecycle policies target only instances that are in the same Region as the policy.
- The first AMI creation operation starts within one hour after the specified start time. Subsequent AMI creation operations start within one hour of their scheduled time.
- When Amazon Data Lifecycle Manager deregisters an AMI, it automatically deletes it backing snapshots.
- Target resource tags are case sensitive.
- If you remove the target tags from an instance that is targeted by a policy, Amazon Data Lifecycle Manager no longer manages existing AMIs in the standard; you must manually delete them if they are no longer needed.
- You can create multiple policies to back up an instance. For example, if an instance has two tags, where tag *A* is the target for policy *A* to create an AMI every 12 hours,and tag *B* is the target for policy *B* to create an AMI every 24 hours, Amazon Data Lifecycle Manager creates AMIs according to the schedules for both policies. Alternatively, you can achieve the same result by creating a single policy that has multiple schedules. For example, you can create a single policy that targets only tag *A*, and specify two schedules — one for every 12 hours and one for every 24 hours.
- New volumes that are attached to a target instance after the policy has been created are automatically included in the backup at the next policy run. All volumes attached to the instance at the time of the policy run are included.
- If you create a policy with a custom cron-based schedule that is configured to create only one AMI, the policy will not automatically deregister that AMI when the retention threshold is reached. You must manually deregister the AMI if it is no longer needed.
- If you create an age-based policy where the retention period is shorter than the creation frequency, Amazon Data Lifecycle Manager will always retain the last AMI until the next one is created. For example, if an age-based policy creates one AMI every month with a retention

period of seven days, Amazon Data Lifecycle Manager will retain each AMI for one month, even though the retention period is seven days.

- For count-based policies, Amazon Data Lifecycle Manager always creates AMIs according to the creation frequency before attempting to deregister the oldest AMI according to the retention policy.
- It can take several hours to successfully deregister an AMI and to delete its associated backing snapshots. If Amazon Data Lifecycle Manager creates the next AMI before the previously created AMI is successfully deregistered, you could temporarily retain a number of AMIs that is greater than your retention count.

The following considerations apply to **terminating instances targeted by a policy:**

- If you terminate an instance that was targeted by a policy with a count-based retention schedule, the policy no longer manages the AMIs that it previously created from the terminated instance. You must manually deregister those earlier AMIs if they are no longer needed.
- If you terminate an instance that was targeted by a policy with an age-based retention schedule, the policy continues to deregister AMIs that were previously created from the terminated instance on the defined schedule, up to, but not including, the last AMI. You must manually deregister the last AMI if it is no longer needed.

The following considerations apply to AMI policies and **AMI deprecation:**

- If you increase the AMI deprecation count for a schedule with count-based retention, the change is applied to all AMIs (existing and new) created by the schedule.
- If you increase the AMI deprecation period for a schedule with age-based retention, the change is applied to new AMIs only. Existing AMIs are not affected.
- If you remove the AMI deprecation rule from a schedule, Amazon Data Lifecycle Manager will not cancel deprecation for AMIs that were previously deprecated by that schedule.
- If you decrease the AMI deprecation count or period for a schedule, Amazon Data Lifecycle Manager will not cancel deprecation for AMIs that were previously deprecated by that schedule.
- If you manually deprecate an AMI that was created by an AMI policy, Amazon Data Lifecycle Manager will not override the deprecation.
- If you manually cancel deprecation for an AMI that was previously deprecated by an AMI policy, Amazon Data Lifecycle Manager will not override the cancellation.

- If an AMI is created by multiple conflicting schedules, and one or more of those schedules do not have an AMI deprecation rule, Amazon Data Lifecycle Manager will not deprecate that AMI.
- If an AMI is created by multiple conflicting schedules, and all of those schedules have an AMI deprecation rule, Amazon Data Lifecycle Manager will use the deprecation rule that results in the latest deprecation date.

The following considerations apply to AMI policies and Recycle Bin:

- If Amazon Data Lifecycle Manager deregisters an AMI and sends it to the Recycle Bin when the policy's retention threshold is reached, and you manually restore that AMI from the Recycle Bin, you must manually deregister the AMI when it is no longer needed. Amazon Data Lifecycle Manager will no longer manage the AMI.
- If you manually deregister an AMI that was created by a policy, and that AMI is in the Recycle Bin when the policy's retention threshold is reached, Amazon Data Lifecycle Manager will not deregister the AMI. Amazon Data Lifecycle Manager does not manage AMIs while they are in the Recycle Bin.

  If the AMI is restored from the Recycle Bin before the policy's retention threshold is reached, Amazon Data Lifecycle Manager will deregister the AMI when the policy's retention threshold is reached.

  If the AMI is restored from the Recycle Bin after the policy's retention threshold is reached, Amazon Data Lifecycle Manager will no longer deregister the AMI. You must manually delete it when it is no longer needed.

The following considerations apply to AMI policies that are in the **error** state:

- For policies with age-based retention schedules, AMIs that are set to expire while the policy is in the `error` state are retained indefinitely. You must deregister the AMIs manually. When you re-enable the policy, Amazon Data Lifecycle Manager resumes deregistering AMIs as their retention periods expire.
- For policies with count-based retention schedules, the policy stops creating and deregistering AMIs while it is in the `error` state. When you re-enable the policy, Amazon Data Lifecycle Manager resumes creating AMIs, and it resumes deregistering AMIs as the retention threshold is met.

The following considerations apply to AMI policies and **disabling AMIs**:

- If you disable an AMI created by Amazon Data Lifecycle Manager, and that AMI is disabled when its retention threshold is reached, Amazon Data Lifecycle Manager will deregister the AMI and delete its associated snapshots.
- If you disable an AMI created by Amazon Data Lifecycle Manager and you manually archive its associated snapshots, and those snapshots are archived when their retention threshold is met, Amazon Data Lifecycle Manager will not delete those snapshots and it will no longer manage them.

The following consideration applies to AMI policies and **AMI deregistration protection**:

- If you manually enable deregistration protection for an AMI that was created by Amazon Data Lifecycle Manager, and it is still enabled when the AMI retention threshold is reached, Amazon Data Lifecycle Manager no longer manages that AMI. You must manually deregister the AMI and delete its underlying snapshots if it is no longer needed.

Additional resources

For more information, see the Automating Amazon EBS snapshot and AMI management using Amazon Data Lifecycle Manager AWS storage blog.

Automate cross-account snapshot copies with Data Lifecycle Manager

Automating cross-account snapshot copies enables you to copy your Amazon EBS snapshots to specific Regions in an isolated account and encrypt those snapshots with an encryption key. This enables you to protect yourself against data loss in the event of your account being compromised.

Automating cross-account snapshot copies involves two accounts:

- **Source account**—The source account is the account that creates and shares the snapshots with the target account. In this account, you must create an EBS snapshot policy that creates snapshots at set intervals and then shares them with other AWS accounts.
- **Target account**—The target account is the account with destination account with which the snapshots are shared, and it is the account that creates copies of the shared snapshots. In this

account, you must create a cross-account copy event policy that automatically copies snapshots that are shared with it by one or more specified source accounts.

> **Note**
>
> Both the source account EBS snapshot policy and the target account cross-account copy event policy must be created in the same AWS Region. The target account can then copy snapshots to different destination Regions as needed.

Topics

Create cross-account snapshot copy policies

To prepare the source and target accounts for cross-account snapshot copying, you need to perform the following steps:

Step 1: Create the EBS snapshot policy (*Source account*)

In the source account, create an EBS snapshot policy that will create the snapshots and share them with the required target accounts.

When you create the policy, ensure that you enable cross-account sharing and that you specify the target AWS accounts with which to share the snapshots. These are the accounts with which the snapshots are to be shared. If you are sharing encrypted snapshots, then you must give the selected target accounts permission to use the KMS key used to encrypt the source volume. For more information, see Step 2: Share the customer managed key (*Source account*).

> **Note**
>
> - Create this policy in the same AWS Region where you will create the target account's cross-account copy event policy in Step 3. Both policies must be in the same Region for cross-account snapshot sharing to work properly.

- You can only share snapshots that are unencrypted or that are encrypted using a customer managed key. You can't share snapshots that are encrypted with the default EBS encryption KMS key. If you share encrypted snapshots, then you must also share the KMS key that was used to encrypt the source volume with the target accounts. For more information, see [Allowing users in other accounts to use a KMS key]() in the *AWS Key Management Service Developer Guide*.

For more information about creating an EBS snapshot policy, see [Create Amazon Data Lifecycle Manager custom policy for EBS snapshots]().

Use one of the following methods to create the EBS snapshot policy.

Step 2: Share the customer managed key (*Source account*)

If you are sharing encrypted snapshots, you must grant the IAM role and the target AWS accounts (that you selected in the previous step) permissions to use the customer managed key that was used to encrypt the source volume.

> **Note**
>
> Perform this step only if you are sharing encrypted snapshots. If you are sharing unencrypted snapshots, skip this step.

Console

1. Open the AWS KMS console at https://console.aws.amazon.com/kms.
2. To change the AWS Region, use the Region selector in the upper-right corner of the page.
3. In the navigation pane, choose **Customer managed key** and then select the KMS key that you need to share with the target accounts.

   Make note of the KMS key ARN, you'll need this later.
4. On the **Key policy** tab, scroll down to the **Key users** section. Choose **Add**, enter the name of the IAM role that you selected in the previous step, and then choose **Add**.
5. On the **Key policy** tab, scroll down to the **Other AWS accounts** section. Choose **Add other AWS accounts**, and then add all of the target AWS accounts that you chose to share the snapshots with in the previous step.

6. Choose **Save changes**.

Command line

Use the [get-key-policy] command to retrieve the key policy that is currently attached to the KMS key.

For example, the following command retrieves the key policy for a KMS key with an ID of 9d5e2b3d-e410-4a27-a958-19e220d83a1e and writes it to a file named snapshotKey.json.

```
$ aws kms get-key-policy \
    --policy-name default \
    --key-id 9d5e2b3d-e410-4a27-a958-19e220d83a1e \
    --query Policy \
    --output text > snapshotKey.json
```

Open the key policy using your preferred text editor. Add the ARN of the IAM role that you specified when you created the snapshot policy and the ARNs of the target accounts with which to share the KMS key.

For example, in the following policy, we added the ARN of the default IAM role, and the ARN of the root account for target account 222222222222.

Tip

To follow the principle of least privilege, do not allow full access to kms:CreateGrant. Instead, use the kms:GrantIsForAWSResource condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

```
{
    "Sid" : "Allow use of the key",
    "Effect" : "Allow",
    "Principal" : {
        "AWS" : [
            "arn:aws:iam::111111111111:role/service-role/
AWSDataLifecycleManagerDefaultRole",
```

```
            "arn:aws:iam::222222222222:root"
        ]
    },
    "Action" : [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:ReEncrypt*",
        "kms:GenerateDataKey*",
        "kms:DescribeKey"
    ],
    "Resource" : "*"
},
{
    "Sid" : "Allow attachment of persistent resources",
    "Effect" : "Allow",
    "Principal" : {
        "AWS" : [
            "arn:aws:iam::111111111111:role/service-role/
AWSDataLifecycleManagerDefaultRole",
            "arn:aws:iam::222222222222:root"
        ]
    },
    "Action" : [
        "kms:CreateGrant",
        "kms:ListGrants",
        "kms:RevokeGrant"
    ],
    "Resource" : "*",
    "Condition" : {
        "Bool" : {
          "kms:GrantIsForAWSResource" : "true"
        }
    }
}
```

Save and close the file. Then use the [put-key-policy](#) command to attach the updated key policy to the KMS key.

```
$ aws kms put-key-policy \
    --policy-name default \
    --key-id 9d5e2b3d-e410-4a27-a958-19e220d83a1e \
    --policy file://snapshotKey.json
```

Step 3: Create cross-account copy event policy (*Target account*)

In the target account, you must create a cross-account copy event policy that will automatically copy snapshots that are shared by the required source accounts.

This policy runs in the target account only when one of the specified source accounts shares snapshot with the account.

> **Note**
>
> Create this policy in the same AWS Region as the source account's EBS snapshot policy created in Step 1. Both policies must be in the same Region for cross-account snapshot sharing to work properly. You can then configure this policy to copy snapshots to different destination Regions as needed.

Use one of the following methods to create the cross-account copy event policy.

Console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**, and then choose **Create lifecycle policy**.
3. On the **Select policy type** screen, choose **Cross-account copy event policy**, and then choose **Next**.
4. For **Policy description**, enter a brief description for the policy.
5. For **Policy tags**, add the tags to apply to the lifecycle policy. You can use these tags to identify and categorize your policies.
6. In the **Event settings** section, define the snapshot sharing event that will cause the policy to run. Do the following:

    a. For **Sharing accounts**, specify the source AWS accounts from which you want to copy the shared snapshots. Choose **Add account**, enter the 12-digit AWS account ID, and then choose **Add**.

    b. For **Filter by description**, enter the required snapshot description using a regular expression. Only snapshots that are shared by the specified source accounts and that have descriptions that match the specified filter are copied by the policy. For more information, see [Specify snapshot description filters](#).

7. For **IAM role**, choose the IAM role that has permissions to perform snapshot copy actions. To use the default role provided by Amazon Data Lifecycle Manager, choose **Default role**. Alternatively, to use a custom IAM role that you previously created, choose **Choose another role** and then select the role to use.

   If you are copying encrypted snapshots, you must grant the selected IAM role permissions to use the encryption KMS key used to encrypt the source volume. Similarly, if you are encrypting the snapshot in the destination Region using a different KMS key, you must grant the IAM role permission to use the destination KMS key. For more information, see Step 4: Allow IAM role to use the required KMS keys (*Target account*).

8. In the **Copy action** section, define the snapshot copy actions that the policy should perform when it is activated. The policy can copy snapshots to up to three Regions. You must specify a separate copy rule for each destination Region. For each rule that you add, do the following:

   a. For **Name**, enter a descriptive name for the copy action.

   b. For **Target Region**, select the Region to which to copy the snapshots.

   c. For **Expire**, specify how long to retain the snapshot copies in the target Region after creation.

   d. To encrypt the snapshot copy, for **Encryption**, select **Enable encryption**. If the source snapshot is encrypted, or if encryption by default is enabled for your account, the snapshot copy is always encrypted, even if you do not enable encryption here. If the source snapshot is unencrypted and encryption by default is not enabled for your account, you can choose to enable or disable encryption. If you enable encryption, but do not specify a KMS key, the snapshots are encrypted using the default encryption KMS key in each destination Region. If you specify a KMS key for the destination Region, you must have access to the KMS key.

9. To add additional snapshot copy actions, choose **Add new Regions**.

10. For **Policy status after creation**, choose **Enable policy** to start the policy runs at the next scheduled time, or **Disable policy** to prevent the policy from running. If you do not enable the policy now, it will not start copying snapshots until you manually enable it after creation.

11. Choose **Create policy**.

Command line

Use the create-lifecycle-policy command to create a policy. To create a cross-account copy event policy, for PolicyType, specify EVENT_BASED_POLICY.

For example, the following command creates a cross-account copy event policy in target account 222222222222. The policy copies snapshots that are shared by source account 111111111111. The policy copies snapshots to sa-east-1 and eu-west-2. Snapshots copied to sa-east-1 are unencrypted and they are retained for 3 days. Snapshots copied to eu-west-2 are encrypted using KMS key 8af79514-350d-4c52-bac8-8985e84171c7 and they are retained for 1 month. The policy uses the default IAM role.

```
$ aws dlm create-lifecycle-policy \
    --description "Copy policy" \
    --state ENABLED \
    --execution-role-arn arn:aws:iam::222222222222:role/service-role/
AWSDataLifecycleManagerDefaultRole \
    --policy-details file://policyDetails.json
```

The following shows the contents of the policyDetails.json file.

```
{
    "PolicyType" : "EVENT_BASED_POLICY",
    "EventSource" : {
        "Type" : "MANAGED_CWE",
        "Parameters": {
            "EventType" : "shareSnapshot",
            "SnapshotOwner": ["111111111111"]
        }
    },
    "Actions" : [{
        "Name" :"Copy Snapshot to Sao Paulo and London",
        "CrossRegionCopy" : [{
            "Target" : "sa-east-1",
             "EncryptionConfiguration" : {
                 "Encrypted" : false
             },
             "RetainRule" : {
             "Interval" : 3,
            "IntervalUnit" : "DAYS"
            }
        },
```

```
        {
            "Target" : "eu-west-2",
            "EncryptionConfiguration" : {
                 "Encrypted" : true,
                 "CmkArn" : "arn:aws:kms:eu-
west-2:222222222222:key/8af79514-350d-4c52-bac8-8985e84171c7"
            },
            "RetainRule" : {
                "Interval" : 1,
                "IntervalUnit" : "MONTHS"
            }
        }]
    }]
}
```

If the request succeeds, the command returns the ID of the newly created policy. The following is example output.

```
{
    "PolicyId": "policy-9876543210abcdef0"
}
```

Step 4: Allow IAM role to use the required KMS keys (*Target account*)

If you are copying encrypted snapshots, you must grant the IAM role (that you selected in the previous step) permissions to use the customer managed key that was used to encrypt the source volume.

> **Note**
>
> Only perform this step if you are copying encrypted snapshots. If you are copying unencrypted snapshots, skip this step.

Use one of the following methods to add the required policies to the IAM role.

Console

1. Open the IAM console at https://console.aws.amazon.com/iam/.

2. In the navigation pane, select **Roles**. Search for and select the IAM role that you selected when you created the cross-account copy event policy in the previous step. If you chose to use the default role, the role is named **AWSDataLifecycleManagerDefaultRole**.
3. Choose **Add inline policy** and then select the **JSON** tab.
4. Replace the existing policy with the following, and specify the ARN of the KMS key that was used to encrypt the source volumes and that was shared with you by the source account in Step 2.

> **Note**
>
> If you are copying from multiple source accounts, then you must specify the corresponding KMS key ARN from each source account.

In the following example, the policy grants the IAM role permission to use KMS key `1234abcd-12ab-34cd-56ef-1234567890ab`, which was shared by source account `111111111111`, and KMS key `4567dcba-23ab-34cd-56ef-0987654321yz`, which exists in target account `222222222222`.

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to `kms:CreateGrant`. Instead, use the `kms:GrantIsForAWSResource` condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:RevokeGrant",
                "kms:CreateGrant",
                "kms:ListGrants"
```

```
            ],
            "Resource": [
                "arn:aws:kms:us-
east-1:111111111111:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "arn:aws:kms:us-
east-1:222222222222:key/4567dcba-23ab-34cd-56ef-0987654321yz"
            ],
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:us-
east-1:111111111111:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "arn:aws:kms:us-
east-1:222222222222:key/4567dcba-23ab-34cd-56ef-0987654321yz"
            ]
        }
    ]
}
```

5. Choose **Review policy**
6. For **Name**, enter a descriptive name for the policy, and then choose **Create policy**.

Command line

Using your preferred text editor, create a new JSON file named `policyDetails.json`. Add the following policy and specify the ARN of the KMS key that was used to encrypt the source volumes and that was shared with you by the source account in Step 2.

> **Note**
>
> If you are copying from multiple source accounts, then you must specify the corresponding KMS key ARN from each source account.

In the following example, the policy grants the IAM role permission to use KMS key `1234abcd-12ab-34cd-56ef-1234567890ab`, which was shared by source account `111111111111`, and KMS key `4567dcba-23ab-34cd-56ef-0987654321yz`, which exists in target account `222222222222`.

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to `kms:CreateGrant`. Instead, use the `kms:GrantIsForAWSResource` condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:RevokeGrant",
                "kms:CreateGrant",
                "kms:ListGrants"
            ],
            "Resource": [
                "arn:aws:kms:us-
east-1:111111111111:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "arn:aws:kms:us-
east-1:222222222222:key/4567dcba-23ab-34cd-56ef-0987654321yz"
            ],
            "Condition": {
                "Bool": {
                    "kms:GrantIsForAWSResource": "true"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "kms:Encrypt",
                "kms:Decrypt",
                "kms:ReEncrypt*",
                "kms:GenerateDataKey*",
                "kms:DescribeKey"
            ],
            "Resource": [
                "arn:aws:kms:us-
east-1:111111111111:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "arn:aws:kms:us-
east-1:222222222222:key/4567dcba-23ab-34cd-56ef-0987654321yz"
            ]
        }
    ]
}
```

Save and close the file. Then use the put-role-policy command to add the policy to the IAM role.

For example

```
$ aws iam put-role-policy \
    --role-name AWSDataLifecycleManagerDefaultRole \
    --policy-name CopyPolicy \
    --policy-document file://AdminPolicy.json
```

Specify snapshot description filters

When you create the snapshot copy policy in the target account, you must specify a snapshot description filter. The snapshot description filter enables you to specify an additional level of filtering that lets you control which snapshots are copied by the policy. This means that a snapshot is only copied by the policy if it is shared by one of the specified source accounts, and it has a snapshot description that matches the specified filter. In other words, if a snapshot is shared by one of the specified course accounts, but it does not have a description that matches the specified filter, it is not copied by the policy.

The snapshot filter description must be specified using a regular expression. It is a mandatory field when creating cross-account copy event policies using the console and the command line. The following are example regular expressions that can be used:

- `.*`—This filter matches all snapshot descriptions. If you use this expression the policy will copy all snapshots that are shared by one of the specified source accounts.
- `Created for policy: policy-0123456789abcdef0.*`—This filter matches only snapshots that are created by a policy with an ID of `policy-0123456789abcdef0`. If you use an expression like this, only snapshots that are shared with your account by one of the specified source accounts, and that have been created by a policy with the specified ID are copied by the policy.
- `.*production.*`—This filter matches any snapshot that has the word `production` anywhere in its description. If you use this expression the policy will copy all snapshots that are shared by one of the specified source accounts and that have the specified text in their description.

Considerations for cross-account snapshot copy policies

The following considerations apply to cross-account copy event policies:

- The source account EBS snapshot policy and the target account cross-account copy event policy must be created in the same AWS Region. After the snapshot is shared, the target account policy can copy the snapshot to different destination Regions as specified in the copy actions.
- You can only copy snapshots that are unencrypted or that are encrypted using a customer managed key.
- You can create a cross-account copy event policy to copy snapshots that are shared outside of Amazon Data Lifecycle Manager.
- If you want to encrypt snapshots in the target account, then the IAM role selected for the cross-account copy event policy must have permission to use the required KMS key.

Additional resources

For more information, see the [Automating copying encrypted Amazon EBS snapshots across AWS accounts] AWS storage blog.

Modify Amazon Data Lifecycle Manager policies

Keep the following in mind when modifying Amazon Data Lifecycle Manager policies:

- If you modify an AMI or snapshot policy by removing its target tags, the volumes or instances with those tags are no longer managed by the policy.
- If you modify a schedule name, the snapshots or AMIs created under the old schedule name are no longer managed by the policy.
- If you modify an age-based retention schedule to use a new time interval, the new interval is used only for new snapshots or AMIs created after the change. The new schedule does not affect the retention schedule of snapshots or AMIs created before the change.
- You cannot change the retention schedule of a policy from count-based to age-based after creation. To make this change, you must create a new policy.
- If you disable a policy with an age-based retention schedule, the snapshots or AMIs that are set to expire while the policy is disabled are retained indefinitely. You must delete the snapshots or deregister the AMIs manually. When you re-enable the policy, Amazon Data Lifecycle Manager resumes deleting snapshots or deregistering AMIs as their retention periods expire.
- If you disable a policy with a count-based retention schedule, the policy stops creating and deleting snapshots or AMIs. When you re-enable the policy, Amazon Data Lifecycle Manager resumes creating snapshots and AMIs, and it resumes deleting snapshots or AMIs as the retention threshold is met.
- If you disable a policy that has a snapshot archiving-enabled policy, snapshots that are in the archive tier at the time of disabling the policy are no longer managed by Amazon Data Lifecycle Manager. You must manually delete the snapshot if they are no longer needed.
- If you enable snapshot archiving on a count-based schedule, the archiving rule applies to all new snapshots that are created and archived by the schedule, and also applies to existing snapshots that were previously created and archived by the schedule.
- If you enable snapshot archiving on an age-based schedule, the archiving rule applies only to new snapshots created after enabling snapshot archiving. Existing snapshots created before enabling snapshot archiving continue to be deleted from their respective storage tiers, according to the schedule set when those snapshots were originally created and archived.
- If you disable snapshot archiving for a count-based schedule, the schedule immediately stops archiving snapshots. Snapshots that were previously archived by the schedule remain in the archive tier and they will not be deleted by Amazon Data Lifecycle Manager.

- If you disable snapshot archiving for an age-based schedule, the snapshots created by the policy and that are scheduled to be archived are permanently deleted at the scheduled archive date and time, as indicated by the `aws:dlm:expirationTime` system tag.
- If you disable snapshot archiving for a schedule, the schedule immediately stops archiving snapshots. Snapshots that were previously archived by the schedule remain in the archive tier and they will not be deleted by Amazon Data Lifecycle Manager.
- If you modify the archive retention count for a count-based schedule, the new retention count includes existing snapshots that were previously archived by the schedule.
- If you modify the archive retention period for an age-based schedule, the new retention period applies only to snapshots that are archived after modifying the retention rule.

Use one of the following procedures to modify a lifecycle policy.

Console

To modify a lifecycle policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**.
3. Select a lifecycle policy from the list.
4. Choose **Actions**, **Modify lifecycle policy**.
5. Modify the policy settings as needed. For example, you can modify the schedule, add or remove tags, or enable or disable the policy.
6. Choose **Modify policy**.

Command line

Use the update-lifecycle-policy command to modify the information in a lifecycle policy. To simplify the syntax, this example references a JSON file, `policyDetailsUpdated.json`, that includes the policy details.

```
aws dlm update-lifecycle-policy \
    --state DISABLED \
    --execution-role-arn
 arn:aws:iam::12345678910:role/AWSDataLifecycleManagerDefaultRole" \
    --policy-details file://policyDetailsUpdated.json
```

The following is an example of the `policyDetailsUpdated.json` file.

```
{
   "ResourceTypes":[
      "VOLUME"
   ],
   "TargetTags":[
      {
         "Key": "costcenter",
         "Value": "120"
      }
   ],
   "Schedules":[
      {
         "Name": "DailySnapshots",
         "TagsToAdd": [
            {
               "Key": "type",
               "Value": "myDailySnapshot"
            }
         ],
         "CreateRule": {
            "Interval": 12,
            "IntervalUnit": "HOURS",
            "Times": [
               "15:00"
            ]
         },
         "RetainRule": {
            "Count" :5
         },
         "CopyTags": false
      }
   ]
}
```

To view the updated policy, use the `get-lifecycle-policy` command. You can see that the state, the value of the tag, the snapshot interval, and the snapshot start time were changed.

Delete Amazon Data Lifecycle Manager policies

Keep the following in mind when deleting Amazon Data Lifecycle Manager policies:

- If you delete a policy, the snapshots or AMIs created by that policy are not automatically deleted. If you no longer need the snapshots or AMIs, you must delete them manually.
- If you delete a policy that has a snapshot archiving-enabled policy, snapshots that are in the archive tier at the time of deleting the policy are no longer managed by Amazon Data Lifecycle Manager. You must manually delete the snapshot if they are no longer needed.
- If you delete a policy with an archive-enabled, age-based schedule, the snapshots created by the policy and that are scheduled to be archived are permanently deleted at the scheduled archive date and time, as indicated by the `aws:dlm:expirationtime` system tag.

Use one of the following procedures to delete a lifecycle policy.

Console

To delete a lifecycle policy

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Elastic Block Store**, **Lifecycle Manager**.
3. Select a lifecycle policy from the list.
4. Choose **Actions**, **Delete lifecycle policy**.
5. When prompted for confirmation, choose **Delete policy**.

Command line

Use the delete-lifecycle-policy command to delete a lifecycle policy and free up the target tags specified in the policy for reuse.

> **Note**
> You can delete snapshots created only by Amazon Data Lifecycle Manager.

```
aws dlm delete-lifecycle-policy --policy-id policy-0123456789abcdef0
```

The Amazon Data Lifecycle Manager API Reference provides descriptions and syntax for each of the actions and data types for the Amazon Data Lifecycle Manager Query API.

Alternatively, you can use one of the AWS SDKs to access the API in a way that's tailored to the programming language or platform that you're using. For more information, see AWS SDKs.

Control access to Amazon Data Lifecycle Manager using IAM

Access to Amazon Data Lifecycle Manager requires credentials. Those credentials must have permissions to access AWS resources, such as instances, volumes, snapshots, and AMIs.

The following IAM permissions are required to use Amazon Data Lifecycle Manager.

> **Note**
>
> - The `ec2:DescribeAvailabilityZones`, `ec2:DescribeRegions`, `kms:ListAliases`, and `kms:DescribeKey` permissions are required for console users only. If console access is not required, you can remove the permissions.
> - The ARN format of the *AWSDataLifecycleManagerDefaultRole* role differs depending on whether it was created using the console or the AWS CLI. If the role was created using the console, the ARN format is `arn:aws:iam::account_id:role/service-role/AWSDataLifecycleManagerDefaultRole`. If the role was created using the AWS CLI, the ARN format is `arn:aws:iam::account_id:role/AWSDataLifecycleManagerDefaultRole`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "dlm:*",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [
                "arn:aws:iam::111122223333:role/
AWSDataLifecycleManagerDefaultRole",
```

```
                "arn:aws:iam::111122223333:role/
AWSDataLifecycleManagerDefaultRoleForAMIManagement",
                "arn:aws:iam::111122223333:role/service-role/
AWSDataLifecycleManagerDefaultRole",
                "arn:aws:iam::111122223333:role/service-role/
AWSDataLifecycleManagerDefaultRoleForAMIManagement"
            ]
        },
        {
            "Effect": "Allow",
            "Action": "iam:ListRoles",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeAvailabilityZones",
                "ec2:DescribeRegions",
                "kms:ListAliases",
                "kms:DescribeKey"
            ],
            "Resource": "*"
        }
    ]
}
```

Permissions for encryption

Consider the following when working with Amazon Data Lifecycle Manager and encrypted resources.

- If the source volume is encrypted, ensure that the Amazon Data Lifecycle Manager default roles (**AWSDataLifecycleManagerDefaultRole** and **AWSDataLifecycleManagerDefaultRoleForAMIManagement**) have permission to use the KMS keys used to encrypt the volume.
- If you enable **Cross Region copy** for unencrypted snapshots or AMIs backed by unencrypted snapshots, and choose to enable encryption in the destination Region, ensure that the default roles have permission to use the KMS key needed to perform the encryption in the destination Region.

- If you enable **Cross Region copy** for encrypted snapshots or AMIs backed by encrypted snapshots, ensure that the default roles have permission to use both the source and destination KMS keys.
- If you enable snapshot archiving for encrypted snapshots, ensure that the Amazon Data Lifecycle Manager default role (**AWSDataLifecycleManagerDefaultRole** has permission to use the KMS key used to encrypt the snapshot.

For more information, see Allowing users in other accounts to use a KMS key in the *AWS Key Management Service Developer Guide*.

For more information, see Changing permissions for a user in the *IAM User Guide*.

AWS managed policies for Amazon Data Lifecycle Manager

An AWS managed policy is a standalone policy that is created and administered by AWS. AWS managed policies are designed to provide permissions for many common use cases. AWS managed policies make it more efficient for you to assign appropriate permissions to users, groups, and roles, than if you had to write the policies yourself.

However, you can't change the permissions defined in AWS managed policies. AWS occasionally updates the permissions defined in an AWS managed policy. When this occurs, the update affects all principal entities (users, groups, and roles) that the policy is attached to.

Amazon Data Lifecycle Manager provides AWS managed policies for common use cases. These policies make it more efficient to define the appropriate permissions and control access to your resources. The AWS managed policies provided by Amazon Data Lifecycle Manager are designed to be attached to roles that you pass to Amazon Data Lifecycle Manager.

Topics

AWSDataLifecycleManagerServiceRole

The **AWSDataLifecycleManagerServiceRole** policy provides appropriate permissions to Amazon Data Lifecycle Manager to create and manage Amazon EBS snapshot policies and cross-account copy event policies.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots",
                "ec2:DeleteSnapshot",
                "ec2:DescribeInstances",
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ec2:EnableFastSnapshotRestores",
                "ec2:DescribeFastSnapshotRestores",
                "ec2:DisableFastSnapshotRestores",
                "ec2:CopySnapshot",
                "ec2:ModifySnapshotAttribute",
                "ec2:DescribeSnapshotAttribute",
                "ec2:ModifySnapshotTier",
                "ec2:DescribeSnapshotTierStatus",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "events:PutRule",
                "events:DeleteRule",
                "events:DescribeRule",
                "events:EnableRule",
                "events:DisableRule",
                "events:ListTargetsByRule",
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": "arn:aws:events:*:*:rule/AwsDataLifecycleRule.managed-
cwe.*"
        }
    ]
}
```

AWSDataLifecycleManagerServiceRoleForAMIManagement

The **AWSDataLifecycleManagerServiceRoleForAMIManagement** policy provides appropriate permissions to Amazon Data Lifecycle Manager to create and manage Amazon EBS-backed AMI policies.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*::image/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeVolumes",
```

```
                "ec2:DescribeSnapshots"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteSnapshot",
            "Resource": "arn:aws:ec2:*::snapshot/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ResetImageAttribute",
                "ec2:DeregisterImage",
                "ec2:CreateImage",
                "ec2:CopyImage",
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:EnableImageDeprecation",
                "ec2:DisableImageDeprecation"
            ],
            "Resource": "arn:aws:ec2:*::image/*"
        }
    ]
}
```

AWSDataLifecycleManagerSSMFullAccess

Provides Amazon Data Lifecycle Manager permission to perform the Systems Manager actions required to run pre and post scripts on all Amazon EC2 instances.

> **Important**
>
> The policy uses the `aws:ResourceTag` condition key to restrict access to specific SSM documents when using pre and post scripts. To allow Amazon Data Lifecycle Manager to

access the SSM documents, you must ensure that your SSM documents are tagged with `DLMScriptsAccess:true`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowSSMReadOnlyAccess",
            "Effect": "Allow",
            "Action": [
                "ssm:GetCommandInvocation",
                "ssm:ListCommands",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Sid": "AllowTaggedSSMDocumentsOnly",
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:DescribeDocument",
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/DLMScriptsAccess": "true"
                }
            }
        },
        {
            "Sid": "AllowSpecificAWSOwnedSSMDocuments",
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:DescribeDocument",
```

```
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/AWSEC2-CreateVssSnapshot",
                "arn:aws:ssm:*:*:document/AWSSystemsManagerSAP-
CreateDLMSnapshotForSAPHANA"
            ]
        },
        {
            "Sid": "AllowAllEC2Instances",
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ]
        }
    ]
}
```

AWS managed policy updates

AWS services maintain and update AWS managed policies. You can't change the permissions in AWS managed policies. Services occasionally add additional permissions to an AWS managed policy to support new features. This type of update affects all identities (users, groups, and roles) where the policy is attached. Services are most likely to update an AWS managed policy when a new feature is launched or when new operations become available. Services do not remove permissions from an AWS managed policy, so policy updates won't break your existing permissions.

The following table provides details about updates to AWS managed policies for Amazon Data Lifecycle Manager since this service began tracking these changes. For automatic alerts about changes to this page, subscribe to the RSS feed on the Document history for the Amazon EBS User Guide.

Change	Description	Date
AWSDataLifecycleManagerServ	Amazon Data Lifecycle Manager added	December 16, 2024

Change	Description	Date
iceRole — Updated the policy permissions.	the `ec2:DescribeAvailabilityZones` action to grant snapshot policies permission to get information about Local Zones.	
AWSDataLifecycleManagerSSMFullAccess — Updated the policy permissions.	Updated the policy to support application-consistent snapshots for SAP HANA using the `AWSSystemsManagerSAP-CreateDLMSnapshotForSAPHANA` SSM document.	November 17, 2023
AWSDataLifecycleManagerSSMFullAccess — Added a new AWS managed policy.	Amazon Data Lifecycle Manager added the AWSDataLifecycleManagerSSMFullAccess AWS managed policy.	November 7, 2023

Change	Description	Date
AWSDataLifecycleManagerServiceRole — Added permissions to support snapshot archiving.	Amazon Data Lifecycle Manager added the `ec2:ModifySnapshotTier` and `ec2:DescribeSnapshotTierStatus` actions to grant snapshot policies permission to archive snapshots and to check the archive status for snapshots.	September 30, 2022

Change	Description	Date
AWSDataLifecycleManagerServiceRoleForAMIManagement — Added permissions to support AMI deprecation.	Amazon Data Lifecycle Manager added the `ec2:EnableImageDeprecation` and `ec2:DisableImageDeprecation` actions to grant EBS-backed AMI policies permission to enable and disable AMI deprecation.	August 23, 2021
Amazon Data Lifecycle Manager started tracking changes	Amazon Data Lifecycle Manager started tracking changes for its AWS managed policies.	August 23, 2021

IAM service roles for Amazon Data Lifecycle Manager

An AWS Identity and Access Management (IAM) role is similar to a user, in that it is an AWS identity with permissions policies that determine what the identity can and can't do in AWS. However, instead of being uniquely associated with one person, a role is intended to be assumable by anyone who needs it. A service role is a role that an AWS service assumes to perform actions on your behalf. As a service that performs backup operations on your behalf, Amazon Data Lifecycle Manager requires that you pass it a role to assume when performing policy operations on your behalf. For more information about IAM roles, see IAM Roles in the *IAM User Guide*.

The role that you pass to Amazon Data Lifecycle Manager must have an IAM policy with the permissions that enable Amazon Data Lifecycle Manager to perform actions associated with policy operations, such as creating snapshots and AMIs, copying snapshots and AMIs, deleting snapshots, and deregistering AMIs. Different permissions are required for each of the Amazon Data Lifecycle Manager policy types. The role must also have Amazon Data Lifecycle Manager listed as a trusted entity, which enables Amazon Data Lifecycle Manager to assume the role.

Topics

Default service roles for Amazon Data Lifecycle Manager

Amazon Data Lifecycle Manager uses the following default service roles:

- **AWSDataLifecycleManagerDefaultRole**—default role for managing snapshots. It trusts only the `dlm.amazonaws.com` service to assume the role and it allows Amazon Data Lifecycle Manager to perform the actions required by snapshot and cross-account snapshot copy policies on your behalf. This role uses the `AWSDataLifecycleManagerServiceRole` AWS managed policy.

> **Note**
>
> The ARN format of the role differs depending on whether it was created using the console or the AWS CLI. If the role was created using the console, the ARN format is `arn:aws:iam::account_id:role/service-role/AWSDataLifecycleManagerDefaultRole`. If the role was created using the AWS CLI, the ARN format is `arn:aws:iam::account_id:role/AWSDataLifecycleManagerDefaultRole`.

- **AWSDataLifecycleManagerDefaultRoleForAMIManagement**—default role for managing AMIs. It trusts only the `dlm.amazonaws.com` service to assume the role and it allows Amazon Data Lifecycle Manager to perform the actions required by EBS-backed AMI policies on your behalf. This role uses the `AWSDataLifecycleManagerServiceRoleForAMIManagement` AWS managed policy.

If you are using the Amazon Data Lifecycle Manager console, Amazon Data Lifecycle Manager automatically creates the **AWSDataLifecycleManagerDefaultRole** service role the first time you create a snapshot or cross-account snapshot copy policy, and it automatically creates the **AWSDataLifecycleManagerDefaultRoleForAMIManagement** service role the first time you create an EBS-backed AMI policy.

If you are not using the console, you can manually create the service roles using the create-default-role command. For `--resource-type`, specify `snapshot` to create AWSDataLifecycleManagerDefaultRole, or `image` to create AWSDataLifecycleManagerDefaultRoleForAMIManagement.

```
$ aws dlm create-default-role --resource-type snapshot|image
```

If you delete the default service roles, and then need to create them again, you can use the same process to recreate them in your account.

Custom service roles for Amazon Data Lifecycle Manager

As an alternative to using the default service roles, you can create custom IAM roles with the required permissions and then select them when you create a lifecycle policy.

To create a custom IAM role

1. Create roles with the following permissions.

   - Permissions required for managing snapshot lifecycle policies

     JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateSnapshot",
                "ec2:CreateSnapshots",
                "ec2:DeleteSnapshot",
                "ec2:DescribeInstances",
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots",
                "ec2:EnableFastSnapshotRestores",
                "ec2:DescribeFastSnapshotRestores",
                "ec2:DisableFastSnapshotRestores",
```

```
                "ec2:CopySnapshot",
                "ec2:ModifySnapshotAttribute",
                "ec2:DescribeSnapshotAttribute",
                "ec2:ModifySnapshotTier",
                "ec2:DescribeSnapshotTierStatus",
                "ec2:DescribeAvailabilityZones"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "events:PutRule",
                "events:DeleteRule",
                "events:DescribeRule",
                "events:EnableRule",
                "events:DisableRule",
                "events:ListTargetsByRule",
                "events:PutTargets",
                "events:RemoveTargets"
            ],
            "Resource": "arn:aws:events:*:*:rule/
AwsDataLifecycleRule.managed-cwe.*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetCommandInvocation",
                "ssm:ListCommands",
                "ssm:DescribeInstanceInformation"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
```

```
                "ssm:DescribeDocument",
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:*:*:document/*"
            ],
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/DLMScriptsAccess": "true"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand",
                "ssm:DescribeDocument",
                "ssm:GetDocument"
            ],
            "Resource": [
                "arn:aws:ssm:*::document/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ssm:SendCommand"
            ],
            "Resource": [
                "arn:aws:ec2:*:*:instance/*"
            ],
            "Condition": {
                "StringNotLike": {
                    "aws:ResourceTag/DLMScriptsAccess": "false"
                }
            }
        }
    ]
}
```

- Permissions required for managing AMI lifecycle policies

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": [
                "arn:aws:ec2:*::snapshot/*",
                "arn:aws:ec2:*::image/*"
            ]
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:DescribeImages",
                "ec2:DescribeInstances",
                "ec2:DescribeImageAttribute",
                "ec2:DescribeVolumes",
                "ec2:DescribeSnapshots"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:DeleteSnapshot",
            "Resource": "arn:aws:ec2:*::snapshot/*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:ResetImageAttribute",
                "ec2:DeregisterImage",
                "ec2:CreateImage",
                "ec2:CopyImage",
                "ec2:ModifyImageAttribute"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:EnableImageDeprecation",
                "ec2:DisableImageDeprecation"
            ],
            "Resource": "arn:aws:ec2:*::image/*"
        }
    ]
}
```

For more information, see [Creating a Role](#) in the *IAM User Guide*.

2. Add a trust relationship to the roles.

   a. In the IAM console, choose **Roles**.

   b. Select the roles that you created, and then choose **Trust relationships**.

   c. Choose **Edit Trust Relationship**, add the following policy, and then choose **Update Trust Policy**.

      JSON

```
{
 "Version":"2012-10-17",
 "Statement": [{
  "Effect": "Allow",
  "Principal": {
   "Service": "dlm.amazonaws.com"
  },
  "Action": "sts:AssumeRole"
 }]
}
```

We recommend that you use the `aws:SourceAccount` and `aws:SourceArn` condition keys to protect yourself against the [confused deputy problem](#). For example, you could add the following condition block to the previous trust policy. The `aws:SourceAccount` is the owner of the lifecycle policy and the `aws:SourceArn` is the ARN of the lifecycle policy. If you don't know the lifecycle policy ID, you can replace that portion of the ARN with a wildcard (*) and then update the trust policy after you create the lifecycle policy.

```
"Condition": {
```

```
        "StringEquals": {
            "aws:SourceAccount": "account_id"
        },
        "ArnLike": {
            "aws:SourceArn": "arn:partition:dlm:region:account_id:policy/policy_id"
        }
}
```

Monitor Amazon Data Lifecycle Manager policies

You can use the following features to monitor the lifecycle of your snapshots and AMIs.

Features

- Console and AWS CLI
- AWS CloudTrail
- Monitor Data Lifecycle Manager policies using EventBridge
- Monitor Data Lifecycle Manager policies using CloudWatch

Console and AWS CLI

You can view your lifecycle policies using the Amazon EC2 console or the AWS CLI. Each snapshot and AMI created by a policy has a timestamp and policy-related tags. You can filter snapshots and AMIs using these tags to verify that your backups are being created as you intend.

AWS CloudTrail

With AWS CloudTrail, you can track user activity and API usage to demonstrate compliance with internal policies and regulatory standards. For more information, see the AWS CloudTrail User Guide.

Monitor Data Lifecycle Manager policies using EventBridge

Amazon EBS and Amazon Data Lifecycle Manager emit events related to lifecycle policy actions. You can use AWS Lambda and Amazon CloudWatch Events to handle event notifications programmatically. Events are emitted on a best effort basis. For more information, see the Amazon EventBridge User Guide.

The following events are available:

> **Note**
>
> No events are emitted for AMI lifecycle policy actions.

- `createSnapshot` — An Amazon EBS event emitted when a `CreateSnapshot` action succeeds or fails. For more information, see [Amazon EventBridge events for Amazon EBS](#).
- `DLM Policy State Change` — An Amazon Data Lifecycle Manager event emitted when a lifecycle policy enters an error state. The event contains a description of what caused the error.

  The following is an example of an event when the permissions granted by the IAM role are insufficient.

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "DLM Policy State Change",
    source": "aws.dlm",
    "account": "123456789012",
    "time": "2018-05-25T13:12:22Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:dlm:us-east-1:123456789012:policy/policy-0123456789abcdef"
    ],
    "detail": {
        "state": "ERROR",
        "cause": "Role provided does not have sufficient permissions",
        "policy_id": "arn:aws:dlm:us-east-1:123456789012:policy/
policy-0123456789abcdef"
    }
}
```

  The following is an example of an event when a limit is exceeded.

```
{
    "version": "0",
    "id": "01234567-0123-0123-0123-0123456789ab",
    "detail-type": "DLM Policy State Change",
    "source": "aws.dlm",
    "account": "123456789012",
    "time": "2018-05-25T13:12:22Z",
```

```
    "region": "us-east-1",
    "resources": [
        "arn:aws:dlm:us-east-1:123456789012:policy/policy-0123456789abcdef"
    ],
    "detail":{
        "state": "ERROR",
        "cause": "Maximum allowed active snapshot limit exceeded",
        "policy_id": "arn:aws:dlm:us-east-1:123456789012:policy/
policy-0123456789abcdef"
    }
}
```

- `DLM Pre Post Script Notification` — An event that is emitted when a pre or post script is initiated, succeeds, or fails.

  The following is an example event when a VSS backup succeeds.

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "DLM Pre Post Script Notification",
    "source": "aws.dlm",
    "account": "123456789012",
    "time": "2023-10-27T22:04:52Z",
    "region": "us-east-1",
    "resources": ["arn:aws:dlm:us-east-1:123456789012:policy/
policy-01234567890abcdef"],
    "detail": {
        "script_stage": "",
        "result": "success",
        "cause": "",
        "policy_id": "arn:aws:dlm:us-east-1:123456789012:policy/
policy-01234567890abcdef",
        "execution_handler": "AWS_VSS_BACKUP",
        "source": "arn:aws:ec2:us-east-1:123456789012:instance/i-01234567890abcdef",
        "resource_type": "EBS_SNAPSHOT",
        "resources": [{
            "status": "pending",
            "resource_id": "arn:aws:ec2:us-east-1::snapshot/snap-01234567890abcdef",
            "source": "arn:aws:ec2:us-east-1:123456789012:volume/
vol-01234567890abcdef"
        }],
        "request_id": "a1b2c3d4-a1b2-a1b2-a1b2-a1b2c3d4e5f6",
```

```
        "start_time": "2023-10-27T22:03:29.370Z",
        "end_time": "2023-10-27T22:04:51.370Z",
        "timeout_time": ""
    }
}
```

Monitor Data Lifecycle Manager policies using CloudWatch

You can monitor your Amazon Data Lifecycle Manager lifecycle policies using CloudWatch, which collects raw data and processes it into readable, near real-time metrics. You can use these metrics to see exactly how many Amazon EBS snapshots and EBS-backed AMIs are created, deleted, and copied by your policies over time. You can also set alarms that watch for certain thresholds, and send notifications or take actions when those thresholds are met.

Metrics are kept for a period of 15 months, so that you can access historical information and gain a better understanding of how your lifecycle policies perform over an extended period.

For more information about Amazon CloudWatch, see the Amazon CloudWatch User Guide.

Topics

Supported metrics

The following Amazon Data Lifecycle Manager metrics are included in the `AWS/EBS` namespace. The metrics differ by policy type.

All metrics can be measured on the `DLMPolicyId` dimension. The most useful statistics are `sum` and `average`, and the unit of measure is `count`.

Choose a tab to view the metrics supported by that policy type.

EBS snapshot policies

Metric	Description
ResourcesTargeted	The number of resources targeted by the tags specified in a snapshot or EBS-backed AMI policy.
SnapshotsCreateStarted	The number of snapshot create actions initiated by a snapshot policy. Each action is recorded only once, even if there are multiple subsequent retries. If a snapshot create action fails, Amazon Data Lifecycle Manager sends a `SnapshotsCreateFailed` metric.
SnapshotsCreateCompleted	The number of snapshots created by a snapshot policy. This includes successful retries within 60 minutes of the scheduled time.
SnapshotsCreateFailed	The number of snapshots that could not be created by a snapshot policy. This includes unsuccessful retries within 60 minutes from the scheduled time.
SnapshotsSharedCompleted	The number of snapshots shared across accounts by a snapshot policy.
SnapshotsDeleteCompleted	The number of snapshots deleted by a snapshot or EBS-backed AMI policy. This metric applies only to snapshots created by the policy. It does not apply to cross-Region snapshot copies created by the policy. This metric includes snapshots that are deleted when an EBS-backed AMI policy deregisters AMIs.
SnapshotsDeleteFailed	The number of snapshots that could not be deleted by a snapshot or EBS-backed AMI policy. This metric applies only to snapshots created by the policy. It does not apply to cross-Region snapshot copies created by the policy.

Metric	Description
	This metric includes snapshots that are deleted when an EBS-backed AMI policy deregisters AMIs.
SnapshotsCopiedRegionStarted	The number of cross-Region snapshot copy actions initiated by a snapshot policy.
SnapshotsCopiedRegionCompleted	The number of cross-Region snapshot copies created by a snapshot policy. This includes successful retries within 24 hours of the scheduled time.
SnapshotsCopiedRegionFailed	The number of cross-Region snapshot copies that could not be created by a snapshot policy. This includes unsuccessful retries within 24 hours from the scheduled time.
SnapshotsCopiedRegionDeleteCompleted	The number of cross-Region snapshot copies deleted, as designated by the retention rule, by a snapshot policy.
SnapshotsCopiedRegionDeleteFailed	The number of cross-Region snapshot copies that could not be deleted, as designated by the retention rule, by a snapshot policy.
snapshotsArchiveDeletionFailed	The number of archived snapshots that could not be deleted from the archive tier by a snapshot policy.
snapshotsArchiveScheduled	The number of snapshots that were scheduled to be archived by a snapshot policy.
snapshotsArchiveCompleted	The number of snapshots that were successfully archived by a snapshot policy.

Metric	Description
snapshotsArchiveFailed	The number of snapshots that could not be archived by a snapshot policy.
snapshotsArchiveDeletionCompleted	The number of archived snapshots that were successfully deleted from the archive tier by a snapshot policy.
PreScriptStarted	The number of instances for which a pre script was successfully initiated. If script retries are enabled, this metric can be emitted multiple times per policy run.
PreScriptCompleted	The number of instances for which a pre script was successfully completed. The metric is emitted even if the pre script completes outside of the specified timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.
PreScriptFailed	The number of instances for which a pre script failed to complete successfully. The metric is emitted even if the pre script completes outside of the specified timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.
PostScriptStarted	The number of instances for which a post script was successfully initiated. If script retries are enabled, this metric can be emitted multiple times per policy run.

Metric	Description
PostScriptCompleted	The number of instances for which a post script was successfully completed. The metric is emitted even if the post script completes outside of the specified timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.
PostScriptFailed	The number of instances for which a post script failed to complete successfully. The metric is emitted even if the post script completes outside of the specified timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.
`VSSBackupStarted`	The number of instances for which a VSS backup was successfully initiated. If script retries are enabled, this metric can be emitted multiple times per policy run.
`VSSBackupCompleted`	The number of instances for which a VSS backup was successfully completed. The metric is emitted even if the VSS backup completes outside of the timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.
`VSSBackupFailed`	The number of instances for which a VSS backup failed to complete successfully. The metric is emitted even if the VSS backup completes outside of the timeout period. If script retries are enabled, this metric can be emitted multiple times per policy run.

EBS-backed AMI policies

The following metrics can be used with EBS-backed AMI policies:

Metric	Description
ResourcesTargeted	The number of resources targeted by the tags specified in a snapshot or EBS-backed AMI policy.
SnapshotsDeleteCompleted	The number of snapshots deleted by a snapshot or EBS-backed AMI policy. This metric applies only to snapshots created by the policy. It does not apply to cross-Region snapshot copies created by the policy. This metric includes snapshots that are deleted when an EBS-backed AMI policy deregisters AMIs.
SnapshotsDeleteFailed	The number of snapshots that could not be deleted by a snapshot or EBS-backed AMI policy. This metric applies only to snapshots created by the policy. It does not apply to cross-Region snapshot copies created by the policy. This metric includes snapshots that are deleted when an EBS-backed AMI policy deregisters AMIs.
SnapshotsCopiedRegionDeleteCompleted	The number of cross-Region snapshot copies deleted, as designated by the retention rule, by a snapshot policy.
SnapshotsCopiedRegionDeleteFailed	The number of cross-Region snapshot copies that could not be deleted, as designated by the retention rule, by a snapshot policy.
ImagesCreateStarted	The number of **CreateImage** actions initiated by an EBS-backed AMI policy.

Metric	Description
`ImagesCreateCompleted`	The number of AMIs created by an EBS-backed AMI policy.
`ImagesCreateFailed`	The number of AMIs that could not be created by an EBS-backed AMI policy.
`ImagesDeregisterCompleted`	The number of AMIs deregistered by an EBS-backed AMI policy.
`ImagesDeregisterFailed`	The number of AMIs that could not be deregistered by an EBS-backed AMI policy.
`ImagesCopiedRegionStarted`	The number of cross-Region copy actions initiated by an EBS-backed AMI policy.
`ImagesCopiedRegionCompleted`	The number of cross-Region AMI copies created by an EBS-backed AMI policy.
`ImagesCopiedRegionFailed`	The number of cross-Region AMI copies that could not be created by an EBS-backed AMI policy.
`ImagesCopiedRegionDeregisterCompleted`	The number of cross-Region AMI copies deregistered, as designated by the retention rule, by an EBS-backed AMI policy.

Metric	Description
ImagesCopiedRegionDeregisteredFailed	The number of cross-Region AMI copies that could not be deregistered, as designated by the retention rule, by an EBS-backed AMI policy.
EnableImageDeprecationCompleted	The number of AMIs that were marked for deprecation by an EBS-backed AMI policy.
EnableImageDeprecationFailed	The number of AMIs that could not be marked for deprecation by an EBS-backed AMI policy.
EnableCopiedImageDeprecationCompleted	The number of cross-Region AMI copies that were marked for deprecation by an EBS-backed AMI policy.
EnableCopiedImageDeprecationFailed	The number of cross-Region AMI copies that could not be marked for deprecation by an EBS-backed AMI policy.

Cross-account copy event policies

The following metrics can be used with cross-account copy event policies:

Metric	Description
	The number of cross-account snapshot copy actions initiated by a cross-account copy event policy.

Metric	Description
SnapshotsCopiedAccountStarted	
SnapshotsCopiedAccountCompleted	The number of snapshots copied from another account by a cross-account copy event policy. This includes successful retries within 24 hours of the scheduled time.
SnapshotsCopiedAccountFailed	The number of snapshots that could not be copied from another account by a cross-account copy event policy. This includes unsuccessful retries within 24 hours of the scheduled time.
SnapshotsCopiedAccountDeleteCompleted	The number of cross-Region snapshot copies deleted, as designated by the retention rule, by a cross-account copy event policy.
SnapshotsCopiedAccountDeleteFailed	The number of cross-Region snapshot copies that could not be deleted, as designated by the retention rule, by a cross-account copy event policy.

View CloudWatch metrics for your policies

You can use the AWS Management Console or the command line tools to list the metrics that Amazon Data Lifecycle Manager sends to Amazon CloudWatch.

Amazon EC2 console

To view metrics using the Amazon EC2 console

1. Open the Amazon EC2 console at https://console.aws.amazon.com/ec2/.
2. In the navigation pane, choose **Lifecycle Manager**.

3. Select a policy in the grid and then choose the **Monitoring** tab.

CloudWatch console

To view metrics using the Amazon CloudWatch console

1. Open the CloudWatch console at https://console.aws.amazon.com/cloudwatch/.
2. In the navigation pane, choose **Metrics**.
3. Select the **EBS** namespace and then select **Data Lifecycle Manager metrics**.

AWS CLI

To list all the available metrics for Amazon Data Lifecycle Manager

Use the list-metrics command.

```
$ C:\> aws cloudwatch list-metrics \
    --namespace AWS/EBS
```

To list all the metrics for a specific policy

Use the list-metrics command and specify the `DLMPolicyId` dimension.

```
$ C:\> aws cloudwatch list-metrics \
    --namespace AWS/EBS \
    --dimensions Name=DLMPolicyId,Value=policy-abcdef01234567890
```

To list a single metric across all policies

Use the list-metrics command and specify the `--metric-name` option.

```
$ C:\> aws cloudwatch list-metrics \
    --namespace AWS/EBS \
    --metric-name SnapshotsCreateCompleted
```

Graph metrics for your policies

After you create a policy, you can open the Amazon EC2 console and view the monitoring graphs for the policy on the **Monitoring** tab. Each graph is based on one of the available Amazon EC2 metrics.

The following graphs metrics are available:

- Resources targeted (based on `ResourcesTargeted`)
- Snapshot creation started (based on `SnapshotsCreateStarted`)
- Snapshot creation completed (based on `SnapshotsCreateCompleted`)
- Snapshot creation failed (based on `SnapshotsCreateFailed`)
- Snapshot sharing completed (based on `SnapshotsSharedCompleted`)
- Snapshot deletion completed (based on `SnapshotsDeleteCompleted`)
- Snapshot deletion failed (based on `SnapshotsDeleteFailed`)
- Snapshot cross-Region copy started (based on `SnapshotsCopiedRegionStarted`)
- Snapshot cross-Region copy completed (based on `SnapshotsCopiedRegionCompleted`)
- Snapshot cross-Region copy failed (based on `SnapshotsCopiedRegionFailed`)
- Snapshot cross-Region copy deletion completed (based on `SnapshotsCopiedRegionDeleteCompleted`)
- Snapshot cross-Region copy deletion failed (based on `SnapshotsCopiedRegionDeleteFailed`)
- Snapshot cross-account copy started (based on `SnapshotsCopiedAccountStarted`)
- Snapshot cross-account copy completed (based on `SnapshotsCopiedAccountCompleted`)
- Snapshot cross-account copy failed (based on `SnapshotsCopiedAccountFailed`)
- Snapshot cross-account copy deletion completed (based on `SnapshotsCopiedAccountDeleteCompleted`)
- Snapshot cross-account copy deletion failed (based on `SnapshotsCopiedAccountDeleteFailed`)
- AMI creation started (based on `ImagesCreateStarted`)
- AMI creation completed (based on `ImagesCreateCompleted`)
- AMI creation failed (based on `ImagesCreateFailed`)
- AMI deregistration completed (based on `ImagesDeregisterCompleted`)

- AMI deregistration failed (based on `ImagesDeregisterFailed`)
- AMI cross-Region copy started (based on `ImagesCopiedRegionStarted`)
- AMI cross-Region copy completed (based on `ImagesCopiedRegionCompleted`)
- AMI cross-Region copy failed (based on `ImagesCopiedRegionFailed`)
- AMI cross-Region copy deregistration completed (based on `ImagesCopiedRegionDeregisterCompleted`)
- AMI cross-Region copy deregister failed (based on `ImagesCopiedRegionDeregisteredFailed`)
- AMI enable deprecation completed (based on `EnableImageDeprecationCompleted`)
- AMI enable deprecation failed (based on `EnableImageDeprecationFailed`)
- AMI cross-Region copy enable deprecation completed (based on `EnableCopiedImageDeprecationCompleted`)
- AMI cross-Region copy enable deprecation failed (based on `EnableCopiedImageDeprecationFailed`)

Create a CloudWatch alarm for a policy

You can create a CloudWatch alarm that monitors CloudWatch metrics for your policies. CloudWatch will automatically send you a notification when the metric reaches a threshold that you specify. You can create a CloudWatch alarm using the CloudWatch console.

For more information about creating alarms using the CloudWatch console, see the following topic in the *Amazon CloudWatch User Guide*.

- Create a CloudWatch Alarm Based on a Static Threshold
- Create a CloudWatch Alarm Based on Anomaly Detection

Example use cases

The following are example use cases.

Topics

- Example 1: ResourcesTargeted metric
- Example 2: SnapshotDeleteFailed metric
- Example 3: SnapshotsCopiedRegionFailed metric

Example 1: ResourcesTargeted metric

You can use the `ResourcesTargeted` metric to monitor the total number of resources that are targeted by a specific policy each time it is run. This enables you to trigger an alarm when the number of targeted resources is below or above an expected threshold.

For example, if you expect your daily policy to create backups of no more than `50` volumes, you can create an alarm that sends an email notification when the `sum` for `ResourcesTargeted` is greater than `50` over a `1` hour period. In this way, you can ensure that no snapshots have been unexpectedly created from volumes that have been incorrectly tagged.

You can use the following command to create this alarm:

```
$ C:\> aws cloudwatch put-metric-alarm \
    --alarm-name resource-targeted-monitor \
    --alarm-description "Alarm when policy targets more than 50 resources" \
    --metric-name ResourcesTargeted \
    --namespace AWS/EBS \
    --statistic Sum \
    --period 3600 \
    --threshold 50 \
    --comparison-operator GreaterThanThreshold \
    --dimensions "Name=DLMPolicyId,Value=policy_id" \
    --evaluation-periods 1 \
    --alarm-actions sns_topic_arn
```

Example 2: SnapshotDeleteFailed metric

You can use the `SnapshotDeleteFailed` metric to monitor for failures to delete snapshots as per the policy's snapshot retention rule.

For example, if you've created a policy that should automatically delete snapshots every twelve hours, you can create an alarm that notifies your engineering team when the `sum` of `SnapshotDeletionFailed` is greater than `0` over a `1` hour period. This could help to investigate improper snapshot retention and to ensure that your storage costs are not increased by unnecessary snapshots.

You can use the following command to create this alarm:

```
$ C:\> aws cloudwatch put-metric-alarm \
    --alarm-name snapshot-deletion-failed-monitor \
```

```
    --alarm-description "Alarm when snapshot deletions fail" \
    --metric-name SnapshotsDeleteFailed \
    --namespace AWS/EBS \
    --statistic Sum \
    --period 3600 \
    --threshold 0 \
    --comparison-operator GreaterThanThreshold \
    --dimensions "Name=DLMPolicyId,Value=policy_id" \
    --evaluation-periods 1 \
    --alarm-actions sns_topic_arn
```

Example 3: SnapshotsCopiedRegionFailed metric

Use the `SnapshotsCopiedRegionFailed` metric to identify when your policies fail to copy snapshots to other Regions.

For example, if your policy copies snapshots across Regions daily, you can create an alarm that sends an SMS to your engineering team when the `sum` of `SnapshotCrossRegionCopyFailed` is greater than `0` over a 1 hour period. This can be useful for verifying whether subsequent snapshots in the lineage were successfully copied by the policy.

You can use the following command to create this alarm:

```
$ C:\> aws cloudwatch put-metric-alarm \
    --alarm-name snapshot-copy-region-failed-monitor \
    --alarm-description "Alarm when snapshot copy fails" \
    --metric-name SnapshotsCopiedRegionFailed \
    --namespace AWS/EBS \
    --statistic Sum \
    --period 3600 \
    --threshold 0 \
    --comparison-operator GreaterThanThreshold \
    --dimensions "Name=DLMPolicyId,Value=policy_id" \
    --evaluation-periods 1 \
    --alarm-actions sns_topic_arn
```

Managing policies that report failed actions

For more information about what to do when one of your policies reports an unexpected non-zero value for a failed action metric, see the article [What should I do if Amazon Data Lifecycle Manager reports failed actions in CloudWatch metrics?]

Service endpoints for Amazon Data Lifecycle Manager

An *endpoint* is a URL that serves as an entry point for an AWS web service. Amazon Data Lifecycle Manager supports the following endpoint types:

- IPv4 endpoints
- Dual-stack endpoints that support both IPv4 and IPv6
- FIPS endpoints

When you make a request, you can specify the endpoint and Region to use. If you do not specify an endpoint, the IPv4 endpoint is used by default. To use a different endpoint type, you must specify it in your request. For examples of how to do this, see Specifying endpoints.

For the Amazon Data Lifecycle Manager, see Amazon Data Lifecycle Manager endpoints in the *Amazon Web Services General Reference*.

Topics

IPv4 endpoints

IPv4 endpoints support IPv4 traffic only. IPv4 endpoints are available for all Regions.

You must specify the Region as part of the endpoint name. The endpoint names use the following naming convention:

- dlm.*region*.amazonaws.com

For example, the IPv4 endpoint for the US East (N. Virginia) Region is `dlm.us-east-1.amazonaws.com`.

Dual-stack (IPv4 and IPv6) endpoints

Dual-stack endpoints support both IPv4 and IPv6 traffic. Dual-stack endpoints are available for all Regions.

To use IPv6, you must use a dual-stack endpoint. When you make a request to a dual-stack endpoint, the endpoint URL resolves to an IPv6 or an IPv4 address, depending on the protocol used by your network and client.

You must specify the Region as part of the endpoint name. Dual-stack endpoint names use the following naming convention:

- `dlm.`*`region`*`.api.aws`

For example, the dual-stack endpoint for the US East (N. Virginia) Region is `dlm.us-east-1.api.aws`.

FIPS endpoints

Amazon Data Lifecycle Manager provides FIPS-validated dual-stack (IPv4 and IPv6) endpoints for the following Regions:

- `us-east-1` — US East (N. Virginia)
- `us-east-2` — US East (Ohio)
- `us-west-1` — US West (N. California)
- `us-west-2` — US West (Oregon)
- `ca-central-1` — Canada (Central)
- `ca-west-1` — Canada West (Calgary)

FIPS dual-stack endpoints use the following naming convention: `dlm-fips.`*`region`*`.api.aws`. For example, the FIPS dual-stack endpoint for the US East (N. Virginia) Region is `dlm-fips.us-east-1.api.aws`.

Specifying endpoints

The following examples show how to specify an endpoint for the `US East (N. Virginia)` Region using the AWS CLI.

- **Dual-stack**

```
aws dlm create-default-role \
--resource-type snapshot \
--endpoint-url https://dlm.us-east-2.api.aws
```

- **IPv4**

```
aws dlm create-default-role \
--resource-type snapshot \
--endpoint-url https://dlm.us-east-2.amazonaws.com
```

Create a private connection between a VPC and Amazon EBS

You can establish a private connection between your VPC and Amazon EBS by creating an *interface VPC endpoint*, powered by AWS PrivateLink. You can access Amazon EBS as if it were in your VPC, without using an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Amazon EBS.

We create an endpoint network interface in each subnet that you enable for the interface endpoint.

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

> **Note**
>
> Amazon Data Lifecycle Manager supports IPv4 interface VPC endpoints for all commercial and AWS GovCloud (US) Regions, and IPv6 interface VPC endpoints for commercial Regions only.

Considerations for Amazon EBS VPC endpoints

Before you set up an interface VPC endpoint for Amazon EBS, review Considerations in the *AWS PrivateLink Guide*.

By default, full access to Amazon EBS is allowed through the endpoint. You can control access to the interface endpoint using VPC endpoint policies. You can attach an endpoint policy to your VPC endpoint that controls access to Amazon EBS. The policy specifies the following information:

- The **principal** that can perform actions.
- The **actions** that can be performed.
- The **resources** on which actions can be performed.

For more information, see [Controlling access to services with VPC endpoints](#) in the *Amazon VPC User Guide*.

The following is an example of an endpoint policy for Amazon EBS. When attached to an endpoint, this policy grants all users permission to get summary information about Amazon Data Lifecycle Manager policies.

```
{
  "Statement": [{
    "Action": "dlm:GetLifecyclePolicies",
    "Effect": "Allow",
    "Principal": "*",
    "Resource": "*"
  }]
}
```

Create an interface VPC endpoint for Amazon EBS

You can create a VPC endpoint for Amazon EBS using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Create a VPC endpoint](#) in the *AWS PrivateLink Guide*.

Create a VPC endpoint for Amazon EBS using the following service name:

- `com.amazonaws.`*`region`*`.dlm`

If you enable private DNS for the endpoint, you can make API requests to Amazon EBS using its default DNS name for the Region, for example, `dlm.us-east-1.amazonaws.com`.

Troubleshoot Amazon Data Lifecycle Manager issues

The following documentation can help you troubleshoot problems that you might encounter.

Topics

Error: `Role with name already exists`

Description

You get the `Role with name AWSDataLifecycleManagerDefaultRole already exists` or `Role with name AWSDataLifecycleManagerDefaultRoleForAMIManagement already exists` error when you try to create a policy using the console.

Cause

The ARN format of the default role differs depending on whether it was created using the console or the AWS CLI. While the ARNs are different, the roles use the same role name, which results in a role naming conflict between the console and the AWS CLI.

Solution

To resolve this issue, do the following:

1. (*For snapshot policies enabled for pre and post scripts only*) Manually attach the **AWSDataLifecycleManagerSSMFullAccess** AWS managed policy to the **AWSDataLifecycleManagerDefaultRole** IAM role. For more information, see Adding IAM identity permissions.
2. When creating your Amazon Data Lifecycle Manager policy, for **IAM role**, select **Choose another role**, and then select either **AWSDataLifecycleManagerDefaultRole** (for a snapshot policy), or **AWSDataLifecycleManagerDefaultRoleForAMIManagement** (for an AMI policy).
3. Continue to create the policy as usual.

Use EBS direct APIs to access the contents of an EBS snapshot

You can use the Amazon Elastic Block Store (Amazon EBS) direct APIs to create EBS snapshots, write data directly to your snapshots, read data on your snapshots, and identify the differences or changes between two snapshots. If you're an independent software vendor (ISV) who offers backup services for Amazon EBS, the EBS direct APIs make it more efficient and cost-effective to track incremental changes on your EBS volumes through snapshots. This can be done without having to create new volumes from snapshots, and then use Amazon Elastic Compute Cloud (Amazon EC2) instances to compare the differences.

You can create incremental snapshots directly from data on-premises into EBS volumes and the cloud to use for quick disaster recovery. With the ability to write and read snapshots, you can write your on-premises data to an EBS snapshot during a disaster. Then after recovery, you can restore it back to AWS or on-premises from the snapshot. You no longer need to build and maintain complex mechanisms to copy data to and from Amazon EBS.

This user guide provides an overview of the elements that make up the EBS direct APIs, and examples of how to use them effectively. For more information about the actions, data types, parameters, and errors of the APIs, see the EBS direct APIs reference. For more information about the supported AWS Regions, endpoints, and service quotas for the EBS direct APIs, see Amazon EBS endpoints and quotas in the *AWS General Reference*.

Topics

- Optimize performance for EBS direct APIs
- Service endpoints for EBS direct APIs
- AWS SDK code examples for EBS direct APIs
- Create a private connection between a VPC and EBS direct APIs
- Log EBS direct APIs calls using AWS CloudTrail
- Frequently asked questions for EBS direct APIs

Pricing for EBS direct APIs

Pricing for APIs

The price that you pay to use the EBS direct APIs depends on the requests you make. For more information, see Amazon EBS pricing.

- **ListChangedBlocks** and ListSnapshotBlocks APIs are charged per request. For example, if you make 100,000 ListSnapshotBlocks API requests in a Region that charges $0.0006 per 1,000 requests, you will be charged $0.06 ($0.0006 per 1,000 requests x 100).
- **GetSnapshotBlock** is charged per block returned. For example, if you make 100,000 GetSnapshotBlock API requests in a Region that charges $0.003 per 1,000 blocks returned, you will be charged $0.30 ($0.003 per 1,000 blocks returned x 100).
- **PutSnapshotBlock** is charged per block written. For example, if you make 100,000 PutSnapshotBlock API requests in a Region that charges $0.006 per 1,000 blocks written, you will be charged $0.60 ($0.006 per 1,000 blocks written x 100).

Networking costs

Data transfer costs

Data transferred directly between EBS direct APIs and Amazon EC2 instances in the same AWS Region is free when using non-FIPS endpoints. For more information, see AWS service endpoints. If other AWS services are in the path of your data transfer, you will be charged their associated data processing costs. These services include, but are not limited to, PrivateLink endpoints, NAT Gateway and Transit Gateway.

VPC interface endpoints

If you are using EBS direct APIs from Amazon EC2 instances or AWS Lambda functions in private subnets, you can use VPC interface endpoints, instead of using NAT gateways, to reduce network data transfer costs. For more information, see Create a private connection between a VPC and EBS direct APIs.

Concepts for EBS direct APIs

The following are the key concepts that you should understand before getting started with the EBS direct APIs.

Snapshots

Snapshots are the primary means to back up data from your EBS volumes. With the EBS direct APIs, you can also back up data from your on-premises disks to snapshots. To save storage costs, successive snapshots are incremental, containing only the volume data that changed since the previous snapshot. For more information, see Amazon EBS snapshots.

> **Note**
>
> EBS direct APIs does not support public snapshots and local snapshots on AWS Outposts.

Blocks

A block is a fragment of data within a snapshot. Each snapshot can contain thousands of blocks. All blocks in a snapshot are of a fixed size.

Block indexes

A block index is a logical index in units of 512 KiB blocks. To identify the block index, divide the logical offset of the data in the logical volume by the block size (logical offset of data/524288). The logical offset of the data must be 512 KiB aligned.

Block tokens

A block token is the identifying hash of a block within a snapshot, and it is used to locate the block data. Block tokens returned by EBS direct APIs are temporary. They change on the expiry timestamp specified for them, or if you run another ListSnapshotBlocks or ListChangedBlocks request for the same snapshot.

Checksum

A checksum is a small-sized datum derived from a block of data for the purpose of detecting errors that were introduced during its transmission or storage. The EBS direct APIs use checksums to validate data integrity. When you read data from an EBS snapshot, the service provides Base64-encoded SHA256 checksums for each block of data transmitted, which you can use for validation. When you write data to an EBS snapshot, you must provide a Base64 encoded SHA256 checksum for each block of data transmitted. The service validates the data received using the checksum provided. For more information, see Use EBS direct APIs checksums to validate snapshot data later in this guide.

Encryption

Encryption protects your data by converting it into unreadable code that can be deciphered only by people who have access to the KMS key used to encrypt it. You can use the EBS direct APIs to read and write encrypted snapshots, but there are some limitations. For more information, see Encryption outcomes for EBS direct APIs later in this guide.

API actions

The EBS direct APIs consists of six actions. There are three read actions and three write actions. The read actions are:

- **ListSnapshotBlocks** — returns the block indexes and block tokens of blocks in the specified snapshot
- **ListChangedBlocks** — returns the block indexes and block tokens of blocks that are different between two specified snapshots of the same volume and snapshot lineage.
- **GetSnapshotBlock** — returns the data in a block for the specified snapshot ID, block index, and block token.

The write actions are:

- **StartSnapshot** — starts a snapshot, either as an incremental snapshot of an existing one or as a new snapshot. The started snapshot remains in a pending state until it is completed using the CompleteSnapshot action.

- **PutSnapshotBlock** — adds data to a started snapshot in the form of individual blocks. You must specify a Base64-encoded SHA256 checksum for the block of data transmitted. The service validates the checksum after the transmission is completed. The request fails if the checksum computed by the service doesn't match what you specified.
- **CompleteSnapshot** — completes a started snapshot that is in a pending state. The snapshot is then changed to a completed state.

Signature Version 4 signing

Signature Version 4 is the process to add authentication information to AWS requests sent by HTTP. For security, most requests to AWS must be signed with an access key, which consists of an access key ID and secret access key. These two keys are commonly referred to as your security credentials. For information about how to obtain credentials for your account, see AWS security credentials.

If you intend to manually create HTTP requests, you must learn how to sign them. When you use the AWS Command Line Interface (AWS CLI) or one of the AWS SDKs to make requests to AWS, these tools automatically sign the requests for you with the access key that you specify when you configure the tools. When you use these tools, you don't need to learn how to sign requests yourself.

For more information, see Signing AWS API requests in the *IAM User Guide*.

Control access to EBS direct APIs using IAM

A user must have the following policies to use the EBS direct APIs. For more information, see Changing permissions for a user.

For more information about the EBS direct APIs resources, actions, and condition context keys for use in IAM permission policies, see Actions, resources, and condition keys for Amazon Elastic Block Store in the *Service Authorization Reference*.

> **Important**
>
> Be cautious when assigning the following policies to users. By assigning these policies, you might give access to a user who is denied access to the same resource through the Amazon EC2 APIs, such as the CopySnapshot or CreateVolume actions.

Permissions to read snapshots

The following policy allows the *read* EBS direct APIs to be used on all snapshots in a specific AWS Region. In the policy, replace *<Region>* with the Region of the snapshot.

The following policy allows the *read* EBS direct APIs to be used on snapshots with a specific key-value tag. In the policy, replace *<Key>* with the key value of the tag, and *<Value>* with the value of the tag.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ebs:ListSnapshotBlocks",
                "ebs:ListChangedBlocks",
                "ebs:GetSnapshotBlock"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "aws:ResourceTag/<Key>": "<Value>"
                }
            }
        }
    ]
}
```

The following policy allows all of the *read* EBS direct APIs to be used on all snapshots in the account only within a specific time range. This policy authorizes use of the EBS direct APIs based on the `aws:CurrentTime` global condition key. In the policy, be sure to replace the date and time range shown with the date and time range for your policy.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ebs:ListSnapshotBlocks",
                "ebs:ListChangedBlocks",
                "ebs:GetSnapshotBlock"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "DateGreaterThan": {
                    "aws:CurrentTime": "2018-05-29T00:00:00Z"
                },
                "DateLessThan": {
                    "aws:CurrentTime": "2020-05-29T23:59:59Z"
                }
            }
        }
    ]
}
```

For more information, see [Changing permissions for a user]() in the *IAM User Guide*.

Permissions to write snapshots

The following policy allows the *write* EBS direct APIs to be used on all snapshots in a specific AWS Region. In the policy, replace *<Region>* with the Region of the snapshot.

The following policy allows the *write* EBS direct APIs to be used on snapshots with a specific key-value tag. In the policy, replace *<Key>* with the key value of the tag, and *<Value>* with the value of the tag.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ebs:StartSnapshot",
                "ebs:PutSnapshotBlock",
                "ebs:CompleteSnapshot"
            ],
            "Resource": "arn:aws:ec2:*::snapshot/*",
            "Condition": {
                "StringEqualsIgnoreCase": {
                    "aws:ResourceTag/<Key>": "<Value>"
                }
            }
        }
    ]
}
```

The following policy allows all of the EBS direct APIs to be used. It also allows the `StartSnapshot` action only if a parent snapshot ID is specified. Therefore, this policy blocks the ability to start new snapshots without using a parent snapshot.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ebs:*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ebs:ParentSnapshot": "arn:aws:ec2:*::snapshot/*"
                }
            }
        }
    ]
}
```

The following policy allows all of the EBS direct APIs to be used. It also allows only the `user` tag key to be created for a new snapshot. This policy also ensures that the user has access to create tags. The `StartSnapshot` action is the only action that can specify tags.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ebs:*",
            "Resource": "*",
            "Condition": {
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "user"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "*"
        }
    ]
}
```

The following policy allows all of the *write* EBS direct APIs to be used on all snapshots in the account only within a specific time range. This policy authorizes use of the EBS direct APIs based on the `aws:CurrentTime` global condition key. In the policy, be sure to replace the date and time range shown with the date and time range for your policy.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ebs:StartSnapshot",
                "ebs:PutSnapshotBlock",
                "ebs:CompleteSnapshot"
            ],
```

```
                "Resource": "arn:aws:ec2:*::snapshot/*",
                "Condition": {
                    "DateGreaterThan": {
                        "aws:CurrentTime": "2018-05-29T00:00:00Z"
                    },
                    "DateLessThan": {
                        "aws:CurrentTime": "2020-05-29T23:59:59Z"
                    }
                }
            }
        ]
}
```

For more information, see Changing permissions for a user in the *IAM User Guide*.

Permissions to use AWS KMS keys

The following policy grants permission to decrypt an encrypted snapshot using a specific KMS key. It also grants permission to encrypt new snapshots using the default KMS key for EBS encryption. In the policy, replace *<Region>* with the Region of the KMS key, *<AccountId>* with the ID of the AWS account of the KMS key, and *<KeyId>* with the ID of the KMS key.

> **Note**
>
> By default, all principals in the account have access to the default AWS managed KMS key for Amazon EBS encryption, and they can use it for EBS encryption and decryption operations. If you are using a customer managed key, you must create a new key policy or modify the existing key policy for the customer managed key to grant the principal access to the customer managed key. For more information, see Key policies in AWS KMS in the *AWS Key Management Service Developer Guide*.

> **Tip**
>
> To follow the principle of least privilege, do not allow full access to `kms:CreateGrant`. Instead, use the `kms:GrantIsForAWSResource` condition key to allow the user to create grants on the KMS key only when the grant is created on the user's behalf by an AWS service, as shown in the following example.

For more information, see Changing permissions for a user in the *IAM User Guide*.

Read Amazon EBS snapshots with EBS direct APIs

The following steps describe how to use the EBS direct APIs to read snapshots:

1. Use the ListSnapshotBlocks action to view all block indexes and block tokens of blocks in a snapshot. Or use the ListChangedBlocks action to view only the block indexes and block tokens of blocks that are different between two snapshots of the same volume and snapshot lineage. These actions help you identify the block tokens and block indexes of blocks for which you might want to get data.
2. Use the GetSnapshotBlock action, and specify the block index and block token of the block for which you want to get data.

> **Note**
>
> You can't use EBS direct APIs with archived snapshots.

The following examples show how to read snapshots using the EBS direct APIs.

Topics

- List blocks in a snapshot
- List blocks that are different between two snapshots
- Get block data from a snapshot

List blocks in a snapshot

AWS CLI

The following list-snapshot-blocks example command returns the block indexes and block tokens of blocks that are in snapshot `snap-0987654321`. The `--starting-block-index` parameter limits the results to block indexes greater than `1000`, and the `--max-results` parameter limits the results to the first `100` blocks.

```
aws ebs list-snapshot-blocks --snapshot-id snap-0987654321 --starting-block-
index 1000 --max-results 100
```

The following example response for the previous command lists the block indexes and block tokens in the snapshot. Use the `get-snapshot-block` command and specify the block index and block token of the block for which you want to get data. The block tokens are valid until the expiry time listed.


```
{
      "Blocks": [
          {
              "BlockIndex": 1001,
              "BlockToken": "AAABAV3/
PNhXOynVdMYHUpPsetaSvjLB1dtIGfbJv5OJ0sX855EzGTWos4a4"
          },
          {
              "BlockIndex": 1002,
              "BlockToken": "AAABATGQIgwr0WwIuqIMjCA/Sy7e/
YoQFZsHejzGNvjKauzNgzeI13YHBfQB"
          },
          {
              "BlockIndex": 1007,
              "BlockToken": "AAABAZ9CTuQtUvp/
dXqRWw4d07eOgTZ3jvn6hiW30W9duM8MiMw6yQayzF2c"
          },
          {
              "BlockIndex": 1012,
              "BlockToken": "AAABAQdzxhw0rVV6PNmsfo/
YRIxo9JPR85XxPf1BLjg0Hec6pygYr6laE1p0"
          },
          {
              "BlockIndex": 1030,
              "BlockToken": "AAABAaYvPax6mv+iGWLdTUjQtFWouQ7Dqz6nSD9L
+CbXnvpkswA6iDID523d"
          },
          {
              "BlockIndex": 1031,
              "BlockToken": "AAABATgWZC0XcFwUKvTJbUXMiSPg59KVxJGL
+BWBClkw6spzCxJVqDVaTskJ"
          },
          ...
      ],
```

```
    "ExpiryTime": 1576287332.806,
    "VolumeSize": 32212254720,
    "BlockSize": 524288
}
```

AWS API

The following ListSnapshotBlocks example request returns the block indexes and block tokens of blocks that are in snapshot `snap-0acEXAMPLEcf41648`. The `startingBlockIndex` parameter limits the results to block indexes greater than `1000`, and the `maxResults` parameter limits the results to the first `100` blocks.

```
GET /snapshots/snap-0acEXAMPLEcf41648/blocks?maxResults=100&startingBlockIndex=1000
 HTTP/1.1
  Host: ebs.us-east-2.amazonaws.com
  Accept-Encoding: identity
  User-Agent: <User agent parameter>
  X-Amz-Date: 20200617T231953Z
  Authorization: <Authentication parameter>
```

The following example response for the previous request lists the block indexes and block tokens in the snapshot. Use the GetSnapshotBlock action and specify the block index and block token of the block for which you want to get data. The block tokens are valid until the expiry time listed.

```
HTTP/1.1 200 OK
  x-amzn-RequestId: d6e5017c-70a8-4539-8830-57f5557f3f27
  Content-Type: application/json
  Content-Length: 2472
  Date: Wed, 17 Jun 2020 23:19:56 GMT
  Connection: keep-alive

  {
      "BlockSize": 524288,
      "Blocks": [
          {
              "BlockIndex": 0,
              "BlockToken": "AAUBAcuWqOCnDNuKle11s7IIX6jp6FYcC/q8oT93913HhvLvA
+3JRrSybp/0"
          },
          {
              "BlockIndex": 1536,
```

```
                "BlockToken":
"AAUBAWudwfmofcrQhGVlLwuRKm2b8ZXPiyrgoykTRC6IU1NbxKWDY1pPjvnV"
            },
            {
                "BlockIndex": 3072,
                "BlockToken":
"AAUBAV7p6pC5fKAC7TokoNCtAnZhqq27u6YEXZ3MwRevBkDjmMx6iuA6tsBt"
            },
            {
                "BlockIndex": 3073,
                "BlockToken":
"AAUBAbqt9zpqBUEvtO2HINAfFaWToOwlPjbIsQOlx6JUN/0+iMQl0NtNbnX4"
            },
            ...
        ],
        "ExpiryTime": 1.59298379649E9,
        "VolumeSize": 3
    }
```

List blocks that are different between two snapshots

Keep the following in mind when making **paginated requests** to list the changed blocks between two snapshots:

- The response can include one or more empty `ChangedBlocks` arrays. For example:
  - Snapshot 1 — full snapshot with 1000 blocks with block indexes `0` - 999.
  - Snapshot 2 — incremental snapshot with only one changed block with block index 999.

  Listing the changed blocks for these snapshots with `StartingBlockIndex = 0` and `MaxResults = 100` returns an empty array of `ChangedBlocks`. You must request the remaining results using `nextToken` until the changed block is returned in the tenth result set, which includes blocks with block indexes `900` - 999.
- The response can skip unwritten blocks in the snapshots. For example:
  - Snapshot 1 — full snapshot with 1000 blocks with block indexes `2000` - 2999.
  - Snapshot 2 — incremental snapshot with only one changed block with block index `2000`.

  Listing the changed blocks for these snapshots with `StartingBlockIndex = 0` and `MaxResults = 100`, the response skips block indexes `0` - 1999 and includes block index `2000`. The response will not include empty `ChangedBlocks` arrays.

AWS CLI

The following [list-changed-blocks](https://docs.aws.amazon.com/cli/latest/reference/ebs/list-changed-blocks.html) example command returns the block indexes and block tokens of blocks that are different between snapshots `snap-1234567890` and `snap-0987654321`. The `--starting-block-index` parameter limits the results to block indexes greater than `0`, and the `--max-results` parameter limits the results to the first `500` blocks..

```
aws ebs list-changed-blocks --first-snapshot-id snap-1234567890 --second-snapshot-id snap-0987654321 --starting-block-index 0 --max-results 500
```

The following example response for the previous command shows that block indexes 0, 6000, 6001, 6002, and 6003 are different between the two snapshots. Additionally, block indexes 6001, 6002, and 6003 exist only in the first snapshot ID specified, and not in the second snapshot ID because there is no second block token listed in the response.

Use the `get-snapshot-block` command and specify the block index and block token of the block for which you want to get data. The block tokens are valid until the expiry time listed.


```
{
      "ChangedBlocks": [
          {
              "BlockIndex": 0,
              "FirstBlockToken": "AAABAVahm9SO60Dyi0ORySzn2ZjGjW/
KN3uygGlS0QOYWesbzBbDnX2dGpmC",
              "SecondBlockToken":
 "AAABAf8o0o6UFi1rDbSZGIRaCEdDyBu9TlvtCQxxoKV8qrUPQP7vcM6iWGSr"
          },
          {
              "BlockIndex": 6000,
              "FirstBlockToken": "AAABAbYSiZvJ0/
R9tz8suI8dSzecLjN4kkazK8inFXVintPkdaVFLfCMQsKe",
              "SecondBlockToken":
 "AAABAZnqTdzFmKRpsaMAsDxviVqEI/3jJzI2crq2eFDCgHmyNf777elD9oVR"
          },
          {
              "BlockIndex": 6001,
              "FirstBlockToken": "AAABASBpSJ2UAD3PLxJnCt6zun4/
T4sU25Bnb8jB5Q6FRXHFqAIAqE04hJoR"
          },
          {
              "BlockIndex": 6002,
```

```
                "FirstBlockToken": "AAABASqX4/
NWjvNceoyMUljcRd0DnwbSwNnes1UkoP62CrQXvn47BY5435aw"
            },
            {
                "BlockIndex": 6003,
                "FirstBlockToken":
 "AAABASmJ0O5JxAOce25rF4P1sdRtyIDsX12tFEDunnePYUKOf4PBROuICb2A"
            },
            ...
        ],
        "ExpiryTime": 1576308931.973,
        "VolumeSize": 32212254720,
        "BlockSize": 524288,
        "NextToken": "AAADARqElNng/sV98CYk/bJDCXeLJmLJHnNSkHvLzVaO0zsPH/QM3Bi3zF//
O6Mdi/BbJarBnp8h"
  }
```

AWS API

The following ListChangedBlocks example request returns the block indexes and block tokens of blocks that are different between snapshots `snap-0acEXAMPLEcf41648` and `snap-0c9EXAMPLE1b30e2f`. The `startingBlockIndex` parameter limits the results to block indexes greater than `0`, and the `maxResults` parameter limits the results to the first `500` blocks.

```
GET /snapshots/snap-0c9EXAMPLE1b30e2f/changedblocks?
firstSnapshotId=snap-0acEXAMPLEcf41648&maxResults=500&startingBlockIndex=0 HTTP/1.1
  Host: ebs.us-east-2.amazonaws.com
  Accept-Encoding: identity
  User-Agent: <User agent parameter>
  X-Amz-Date: 20200617T232546Z
  Authorization: <Authentication parameter>
```

The following example response for the previous request shows that block indexes `0`, `3072`, `6002`, and `6003` are different between the two snapshots. Additionally, block indexes `6002`, and `6003` exist only in the first snapshot ID specified, and not in the second snapshot ID because there is no second block token listed in the response.

Use the `GetSnapshotBlock` action and specify the block index and block token of the block for which you want to get data. The block tokens are valid until the expiry time listed.

```
HTTP/1.1 200 OK
```

```
x-amzn-RequestId: fb0f6743-6d81-4be8-afbe-db11a5bb8a1f
Content-Type: application/json
Content-Length: 1456
Date: Wed, 17 Jun 2020 23:25:47 GMT
Connection: keep-alive

{
    "BlockSize": 524288,
    "ChangedBlocks": [
        {
            "BlockIndex": 0,
            "FirstBlockToken": "AAUBAVaWqOCnDNuKle11s7IIX6jp6FYcC/
tJuVT1GgP23AuLntwiMdJ+OJkL",
            "SecondBlockToken": "AAUBASxzy0Y0b33JVRLoYm3NOresCxn5RO+HVFzXW3Y/
RwfFaPX2Edx8QHCh"
        },
        {
            "BlockIndex": 3072,
            "FirstBlockToken":
 "AAUBAcHp6pC5fKAC7TokoNCtAnZhqq27u6fxRfZOLEmeXLmHBf2R/Yb24MaS",
            "SecondBlockToken":
 "AAUBARGCaufCqBRZC8tEkPYGGkSv3vqvOjJ2xKDi3ljDFiytUxBLXYgTmkid"
        },
        {
            "BlockIndex": 6002,
            "FirstBlockToken": "AAABASqX4/
NWjvNceoyMUljcRd0DnwbSwNnes1UkoP62CrQXvn47BY5435aw"
        },
        {
            "BlockIndex": 6003,
            "FirstBlockToken":
 "AAABASmJ0O5JxAOce25rF4P1sdRtyIDsX12tFEDunnePYUKOf4PBROuICb2A"
        },
        ...
    ],
    "ExpiryTime": 1.592976647009E9,
    "VolumeSize": 3
}
```

Get block data from a snapshot

AWS CLI

The following get-snapshot-block example command returns the data in the block index 6001 with block token AAABASBpSJ2UAD3PLxJnCt6zun4/T4sU25Bnb8jB5Q6FRXHFqAIAqE04hJoR, in snapshot snap-1234567890. The binary data is output to the data file in the C:\Temp directory on a Windows computer. If you run the command on a Linux or Unix computer, replace the output path with /tmp/data to output the data to the data file in the /tmp directory.

```
aws ebs get-snapshot-block --snapshot-id snap-1234567890 --block-index 6001 --block-token AAABASBpSJ2UAD3PLxJnCt6zun4/T4sU25Bnb8jB5Q6FRXHFqAIAqE04hJoR C:/Temp/data
```

The following example response for the previous command shows the size of the data returned, the checksum to validate the data, and the algorithm of the checksum. The binary data is automatically saved to the directory and file you specified in the request command.

```
{
      "DataLength": "524288",
      "Checksum": "cf0Y6/Fn0oFa4VyjQPOa/iD0zhTflPTKzxGv2OKowXc=",
      "ChecksumAlgorithm": "SHA256"
  }
```

AWS API

The following GetSnapshotBlock example request returns the data in the block index 3072 with block token AAUBARGCaufCqBRZC8tEkPYGGkSv3vqvOjJ2xKDi3ljDFiytUxBLXYgTmkid, in snapshot snap-0c9EXAMPLE1b30e2f.

```
GET /snapshots/snap-0c9EXAMPLE1b30e2f/blocks/3072?blockToken=AAUBARGCaufCqBRZC8tEkPYGGkSv3vqvOjJ2xKDi3ljDFiytUxBLXYgTmkid HTTP/1.1
  Host: ebs.us-east-2.amazonaws.com
  Accept-Encoding: identity
  User-Agent: <User agent parameter>
  X-Amz-Date: 20200617T232838Z
  Authorization: <Authentication parameter>
```

The following example response for the previous request shows the size of the data returned, the checksum to validate the data, and the algorithm used to generate the checksum. The binary data is transmitted in the body of the response and is represented as *BlockData* in the following example.

```
HTTP/1.1 200 OK
  x-amzn-RequestId: 2d0db2fb-bd88-474d-a137-81c4e57d7b9f
  x-amz-Data-Length: 524288
  x-amz-Checksum: Vc0yY2j3qg8bUL9I6GQuI2orTudrQRBDMIhcy7bdEsw=
  x-amz-Checksum-Algorithm: SHA256
  Content-Type: application/octet-stream
  Content-Length: 524288
  Date: Wed, 17 Jun 2020 23:28:38 GMT
  Connection: keep-alive

  BlockData
```

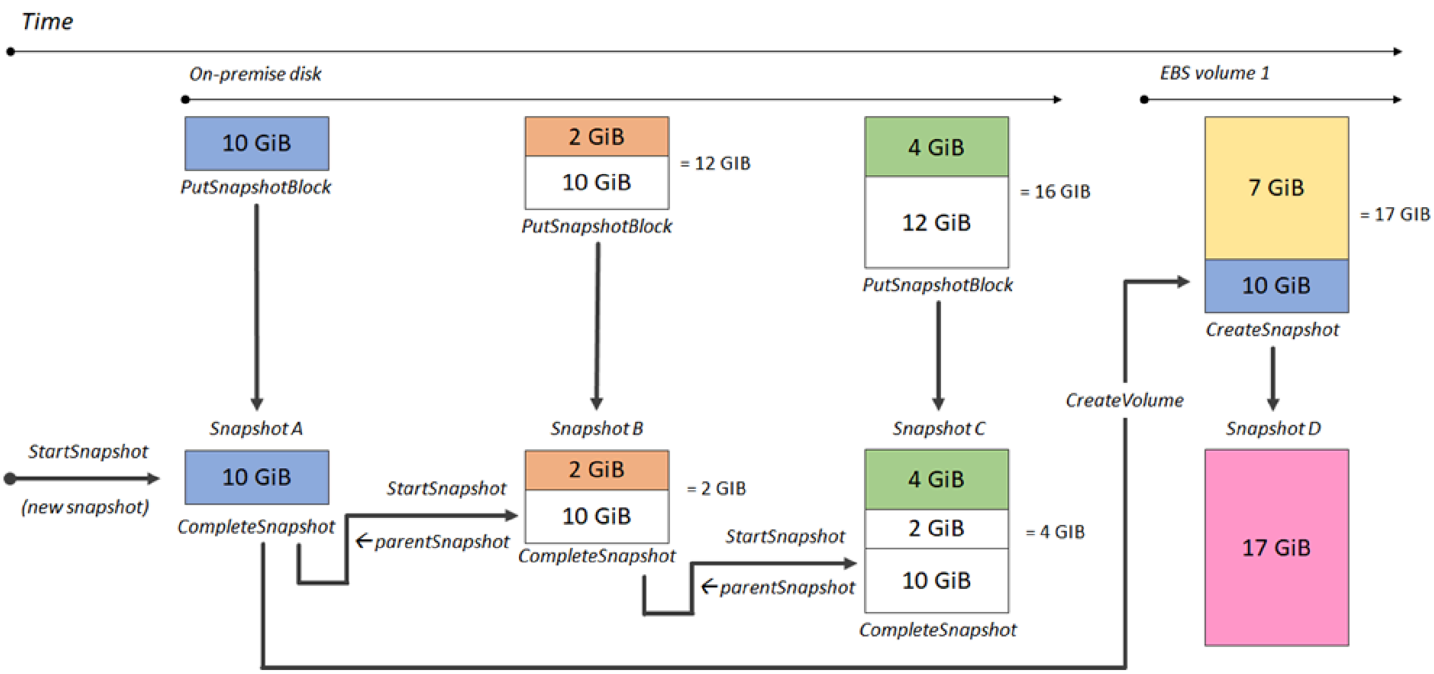
Write Amazon EBS snapshots with EBS direct APIs

The following steps describe how to use the EBS direct APIs to write incremental snapshots:

1. Use the StartSnapshot action and specify a parent snapshot ID to start a snapshot as an incremental snapshot of an existing one, or omit the parent snapshot ID to start a new snapshot. This action returns the new snapshot ID, which is in a pending state.
2. Use the PutSnapshotBlock action and specify the ID of the pending snapshot to add data to it in the form of individual blocks. You must specify a Base64-encoded SHA256 checksum for the block of data transmitted. The service computes the checksum of the data received and validates it with the checksum that you specified. The action fails if the checksums don't match.
3. When you're done adding data to the pending snapshot, use the CompleteSnapshot action to start an asynchronous workflow that seals the snapshot and moves it to a completed state.

Repeat these steps to create a new, incremental snapshot using the previously created snapshot as the parent.

For example, in the following diagram, snapshot A is the first new snapshot started. Snapshot A is used as the parent snapshot to start snapshot B. Snapshot B is used as the parent snapshot to start and create snapshot C. Snapshots A, B, and C are incremental snapshots. Snapshot A is used to create EBS volume 1. Snapshot D is created from EBS volume 1. Snapshot D is an incremental snapshot of A; it is not an incremental snapshot of B or C.



The following examples show how to write snapshots using the EBS direct APIs.

Topics

- [Start a snapshot]
- [Put data into a snapshot]
- [Complete a snapshot]

Start a snapshot

AWS CLI

The following start-snapshot example command starts an 8 GiB snapshot, using snapshot `snap-123EXAMPLE1234567` as the parent snapshot. The new snapshot will be an incremental snapshot of the parent snapshot. The snapshot moves to an error state if there are no put or complete requests made for the snapshot within the specified `60` minute timeout period. The `550e8400-e29b-41d4-a716-446655440000` client token ensures idempotency for the request. If the client token is omitted, the AWS SDK automatically generates one for you. For more information about idempotency, see Ensure idempotency in StartSnapshot API requests.

```
aws ebs start-snapshot --volume-size 8 --parent-snapshot snap-123EXAMPLE1234567 --
timeout 60 --client-token 550e8400-e29b-41d4-a716-446655440000
```

The following example response for the previous command shows the snapshot ID, AWS account ID, status, volume size in GiB, and size of the blocks in the snapshot. The snapshot is started in a `pending` state. Specify the snapshot ID in subsequent `put-snapshot-block` commands to write data to the snapshot, then use the `complete-snapshot` command to complete the snapshot and change its status to `completed`.

```
{
    "SnapshotId": "snap-0aaEXAMPLEe306d62",
    "OwnerId": "111122223333",
    "Status": "pending",
    "VolumeSize": 8,
    "BlockSize": 524288
}
```

AWS API

The following [StartSnapshot](#) example request starts an `8` GiB snapshot, using snapshot `snap-123EXAMPLE1234567` as the parent snapshot. The new snapshot will be an incremental snapshot of the parent snapshot. The snapshot moves to an error state if there are no put or complete requests made for the snapshot within the specified `60` minute timeout period. The `550e8400-e29b-41d4-a716-446655440000` client token ensures idempotency for the request. If the client token is omitted, the AWS SDK automatically generates one for you. For more information about idempotency, see [Ensure idempotency in StartSnapshot API requests](#).

```
POST /snapshots HTTP/1.1
Host: ebs.us-east-2.amazonaws.com
Accept-Encoding: identity
User-Agent: <User agent parameter>
X-Amz-Date: 20200618T040724Z
Authorization: <Authentication parameter>

{
    "VolumeSize": 8,
    "ParentSnapshot": snap-123EXAMPLE1234567,
    "ClientToken": "550e8400-e29b-41d4-a716-446655440000",
    "Timeout": 60
}
```

The following example response for the previous request shows the snapshot ID, AWS account ID, status, volume size in GiB, and size of the blocks in the snapshot. The snapshot is started in a pending state. Specify the snapshot ID in a subsequent `PutSnapshotBlocks` request to write data to the snapshot.

```
HTTP/1.1 201 Created
x-amzn-RequestId: 929e6eb9-7183-405a-9502-5b7da37c1b18
Content-Type: application/json
Content-Length: 181
Date: Thu, 18 Jun 2020 04:07:29 GMT
Connection: keep-alive

{
    "BlockSize": 524288,
    "Description": null,
    "OwnerId": "138695307491",
    "Progress": null,
    "SnapshotId": "snap-052EXAMPLEc85d8dd",
    "StartTime": null,
    "Status": "pending",
    "Tags": null,
    "VolumeSize": 8
}
```

Put data into a snapshot

AWS CLI

The following put-snapshot-block example command writes `524288` Bytes of data to block index `1000` on snapshot `snap-0aaEXAMPLEe306d62`. The Base64 encoded `QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM=` checksum was generated using the `SHA256` algorithm. The data that is transmitted is in the `/tmp/data` file.

```
aws ebs put-snapshot-block --snapshot-id snap-0aaEXAMPLEe306d62
 --block-index 1000 --data-length 524288 --block-data /tmp/data --
checksum QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM= --checksum-algorithm SHA256
```

The following example response for the previous command confirms the data length, checksum, and checksum algorithm for the data received by the service.

```
{
    "DataLength": "524288",
```

```
    "Checksum": "QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM=",
    "ChecksumAlgorithm": "SHA256"
}
```

AWS API

The following PutSnapshot example request writes 524288 Bytes of data to block index 1000 on snapshot snap-052EXAMPLEc85d8dd. The Base64 encoded QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM= checksum was generated using the SHA256 algorithm. The data is transmitted in the body of the request and is represented as *BlockData* in the following example.

```
PUT /snapshots/snap-052EXAMPLEc85d8dd/blocks/1000 HTTP/1.1
Host: ebs.us-east-2.amazonaws.com
Accept-Encoding: identity
x-amz-Data-Length: 524288
x-amz-Checksum: QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM=
x-amz-Checksum-Algorithm: SHA256
User-Agent: <User agent parameter>
X-Amz-Date: 20200618T042215Z
X-Amz-Content-SHA256: UNSIGNED-PAYLOAD
Authorization: <Authentication parameter>

          BlockData
```

The following is example response for the previous request confirms the data length, checksum, and checksum algorithm for the data received by the service.

```
HTTP/1.1 201 Created
x-amzn-RequestId: 643ac797-7e0c-4ad0-8417-97b77b43c57b
x-amz-Checksum: QOD3gmEQOXATfJx2Aa34W4FU2nZGyXfqtsUuktOw8DM=
x-amz-Checksum-Algorithm: SHA256
Content-Type: application/json
Content-Length: 2
Date: Thu, 18 Jun 2020 04:22:12 GMT
Connection: keep-alive

{}
```

Complete a snapshot

AWS CLI

The following [complete-snapshot](#) example command completes snapshot `snap-0aaEXAMPLEe306d62`. The command specifies that 5 blocks were written to the snapshot. The `6D3nmwi5f2F0wlh7xX8QprrJBFzDX8aacdOcA3KCM3c=` checksum represents the checksum for the complete set of data written to a snapshot. For more information about checksums, see [Use EBS direct APIs checksums to validate snapshot data](#) earlier in this guide.

```
aws ebs complete-snapshot --snapshot-id snap-0aaEXAMPLEe306d62 --changed-blocks-count 5 --checksum 6D3nmwi5f2F0wlh7xX8QprrJBFzDX8aacdOcA3KCM3c= --checksum-algorithm SHA256 --checksum-aggregation-method LINEAR
```

The following is an example response for the previous command.

```
{
    "Status": "pending"
}
```

AWS API

The following [CompleteSnapshot](#) example request completes snapshot `snap-052EXAMPLEc85d8dd`. The command specifies that 5 blocks were written to the snapshot. The `6D3nmwi5f2F0wlh7xX8QprrJBFzDX8aacdOcA3KCM3c=` checksum represents the checksum for the complete set of data written to a snapshot.

```
POST /snapshots/completion/snap-052EXAMPLEc85d8dd HTTP/1.1
Host: ebs.us-east-2.amazonaws.com
Accept-Encoding: identity
x-amz-ChangedBlocksCount: 5
x-amz-Checksum: 6D3nmwi5f2F0wlh7xX8QprrJBFzDX8aacdOcA3KCM3c=
x-amz-Checksum-Algorithm: SHA256
x-amz-Checksum-Aggregation-Method: LINEAR
User-Agent: <User agent parameter>
X-Amz-Date: 20200618T043158Z
Authorization: <Authentication parameter>
```

The following is an example response for the previous request.

```
HTTP/1.1 202 Accepted
x-amzn-RequestId: 06cba5b5-b731-49de-af40-80333ac3a117
Content-Type: application/json
Content-Length: 20
Date: Thu, 18 Jun 2020 04:31:50 GMT
Connection: keep-alive

{"Status":"pending"}
```

Encryption outcomes for EBS direct APIs

When you start a new snapshot using StartSnapshot, the encryption status depends on the values that you specify for **Encrypted**, **KmsKeyArn**, and **ParentSnapshotId**, and whether your AWS account is enabled for encryption by default.

> **Note**
>
> - You might need additional IAM permissions to use the EBS direct APIs with encryption. For moreinformation, see Permissions to use AWS KMS keys.
> - If Amazon EBS encryption by default is enabled on your AWS account, you can't create unencrypted snapshots.
> - If Amazon EBS encryption by default is enabled on your AWS account, you cannot start a new snapshot using an unencrypted parent snapshot. You must first encrypt the parent snapshot by copying it. For more information, see Copy an Amazon EBS snapshot.

Topics

- Encryption outcomes: Unencrypted parent snapshot
- Encryption outcomes: Encrypted parent snapshot
- Encryption outcomes: No parent snapshot

Encryption outcomes: Unencrypted parent snapshot

The following table describes the encryption outcome for each possible combination of settings when specifying an unencrypted parent snapshot.

ParentSnapshotId	Encrypted	KmsKeyArn	Encryption by default	Result
Unencrypted	Omitted	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	The snapshot is unencrypted.
		Specified	Enabled	
			Disabled	
Unencrypted	True	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	
		Specified	Enabled	
			Disabled	
Unencrypted	False	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	
		Specified	Enabled	
			Disabled	

Encryption outcomes: Encrypted parent snapshot

The following table describes the encryption outcome for each possible combination of settings when specifying an encrypted parent snapshot.

ParentSnapshotId	Encrypted	KmsKeyArn	Encryption by default	Result
Encrypted	Omitted	Omitted	Enabled	The snapshot is encrypted using the same KMS key as the parent snapshot.
			Disabled	

ParentSnapshotId	Encrypted	KmsKeyArn	Encryption by default	Result
		Specified	Enabled	The request fails with `ValidationException` .
			Disabled	
Encrypted	True	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	
		Specified	Enabled	
			Disabled	
Encrypted	False	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	
		Specified	Enabled	
			Disabled	

Encryption outcomes: No parent snapshot

The following tables describe the encryption outcome for each possible combination of settings when not using a parent snapshot.

ParentSnapshotId	Encrypted	KmsKeyArn	Encryption by default	Result
Omitted	True	Omitted	Enabled	The snapshot is encrypted using the default KMS key for your account. *
			Disabled	
		Specified	Enabled	The snapshot is encrypted using the KMS key specified for **KmsKeyArn**.
			Disabled	

ParentSnapshotId	Encrypted	KmsKeyArn	Encryption by default	Result
Omitted	False	Omitted	Enabled	The request fails with `ValidationException` .
			Disabled	The snapshot is unencrypted.
		Specified	Enabled	The request fails with `ValidationException` .
			Disabled	
Omitted	Omitted	Omitted	Enabled	The snapshot is encrypted using the default KMS key for your account. *
			Disabled	The snapshot is unencrypted.
		Specified	Enabled	The snapshot is encrypted using the KMS key specified for **KmsKeyArn**.
			Disabled	

* This default KMS key could be a customer managed key or the default AWS managed KMS key for Amazon EBS encryption.

Use EBS direct APIs checksums to validate snapshot data

The GetSnapshotBlock action returns data that is in a block of a snapshot, and the PutSnapshotBlock action adds data to a block in a snapshot. The block data that is transmitted is not signed as part of the Signature Version 4 signing process. As a result, checksums are used to validate the integrity of the data as follows:

- When you use the GetSnapshotBlock action, the response provides a Base64-encoded SHA256 checksum for the block data using the **x-amz-Checksum** header, and the checksum algorithm using the **x-amz-Checksum-Algorithm** header. Use the returned checksum to validate the integrity of the data. If the checksum that you generate doesn't match what Amazon EBS provided, you should consider the data not valid and retry your request.

- When you use the PutSnapshotBlock action, your request must provide a Base64-encoded SHA256 checksum for the block data using the **x-amz-Checksum** header, and the checksum algorithm using the **x-amz-Checksum-Algorithm** header. The checksum that you provide is validated against a checksum generated by Amazon EBS to validate the integrity of the data. If the checksums do not correspond, the request fails.
- When you use the CompleteSnapshot action, your request can optionally provide an aggregate Base64-encoded SHA256 checksum for the complete set of data added to the snapshot. Provide the checksum using the **x-amz-Checksum** header, the checksum algorithm using the **x-amz-Checksum-Algorithm** header, and the checksum aggregation method using the **x-amz-Checksum-Aggregation-Method** header. To generate the aggregated checksum using the linear aggregation method, arrange the checksums for each written block in ascending order of their block index, concatenate them to form a single string, and then generate the checksum on the entire string using the SHA256 algorithm.

The checksums in these actions are part of the Signature Version 4 signing process.

Ensure idempotency in StartSnapshot API requests

Idempotency ensures that an API request completes only once. With an idempotent request, if the original request completes successful, the subsequent retries return the result from the original successful request and they have no additional effect.

The StartSnapshot API supports idempotency using a *client token*. A client token is a unique string that you specify when you make an API request. If you retry an API request with the same client token and the same request parameters after it has completed successfully, the result of the original request is returned. If you retry a request with the same client token, but change one or more of the request parameters, the `ConflictException` error is returned.

If you do not specify your own client token, the AWS SDKs automatically generates a client token for the request to ensure that it is idempotent.

A client token can be any string that includes up to 64 ASCII characters. You should not reuse the same client tokens for different requests.

To make an idempotent StartSnapshot request with your own client token using the API

Specify the `ClientToken` request parameter.

```
POST /snapshots HTTP/1.1
Host: ebs.us-east-2.amazonaws.com
Accept-Encoding: identity
User-Agent: <User agent parameter>
X-Amz-Date: 20200618T040724Z
Authorization: <Authentication parameter>

{
    "VolumeSize": 8,
    "ParentSnapshot": snap-123EXAMPLE1234567,
    "ClientToken": "550e8400-e29b-41d4-a716-446655440000",
    "Timeout": 60
}
```

To make an idempotent StartSnapshot request with your own client token using the AWS CLI

Specify the `client-token` request parameter.

```
$ C:\> aws ebs start-snapshot --region us-east-2 --volume-size 8 --parent-snapshot snap-123EXAMPLE1234567 --timeout 60 --client-token 550e8400-e29b-41d4-a716-446655440000
```

Error retries for EBS direct APIs

The **AWS SDKs** implement automatic retry logic for requests that return error responses. You can configure the retry settings for the AWS SDKs. For more information, see your SDK's documentation.

You can configure the **AWS CLI** to automatically retry some failed requests. For more information about configuring retries for the AWS CLI, see AWS CLI retries in the *AWS Command Line Interface User Guide*.

The **AWS Query API** does not support retry logic for failed requests. If you are using HTTP or HTTPS requests, you must implement retry logic in your client application.

The following table shows the possible API error responses. Some API errors are retryable. Your client application should always retry failed requests that receive a retryable error.

Error	Response code	Description	Thrown by	Retryable?
`InternalServerException`	500	The request failed due to a network or AWS server-side issue.	All APIs	Yes
`ThrottlingException`	400	The number of API requests has exceeded the maximum allowed API request throttling limit for the account.	All APIs	Yes
`RequestThrottleException`	400	The number of API requests has exceeded the maximum allowed API request throttling limit for the snapshot.	GetSnapshotBlock | PutSnapshotBlock	Yes
`ValidationException` with message `"Failed to read block data"`	400	The provided data block was not readable.	PutSnapshotBlock	Yes
`ValidationException` with any other message	400	The request syntax is malformed, or the input does not satisfy	All APIs	No

Error	Response code	Description	Thrown by	Retryable?
		the constraints specified by the AWS service.		
`ResourceNotFoundException`	404	The specified snapshot ID does not exist.	All APIs	No
`ConflictException`	409	The specified client token was previously used in a similar request that had different request parameters. For more information, see [Ensure idempotency in StartSnapshot API requests]().	StartSnapshot	No
`AccessDeniedException`	403	You do not have permission to perform the requested operation.	All APIs	No
`ServiceQuotaExceededException`	402	The request failed because fulfilling the request would exceed one or more dependent service quotas for your account.	All APIs	No

Error	Response code	Description	Thrown by	Retryable?
`InvalidSignatureException`	403	The request authorization signature has expired. You can retry the request only after refreshing the authorization signature.	All APIs	No

Optimize performance for EBS direct APIs

You can run API requests concurrently. Assuming PutSnapshotBlock latency is 100ms, then a thread can process 10 requests in one second. Furthermore, assuming your client application creates multiple threads and connections (for example, 100 connections), it can make 1000 (10 * 100) requests per second in total. This will correspond to a throughput of around 500 MB per second.

The following list contains few things to look for in your application:

- Is each thread using a separate connection? If the connections are limited on the application then multiple threads will wait for the connection to be available and you will notice lower throughput.
- Is there any wait time in the application between two put requests? This will reduce the effective throughput of a thread.
- The bandwidth limit on the instance – If bandwidth on the instance is shared by other applications, it could limit the available throughput for PutSnapshotBlock requests.

Be sure to take note of other workloads that might be running in the account to avoid bottlenecks. You should also build retry mechanisms into your EBS direct APIs workflows to handle throttling, timeouts, and service unavailability.

Review the EBS direct APIs service quotas to determine the maximum API requests that you can run per second. For more information, see Amazon Elastic Block Store Endpoints and Quotas in the *AWS General Reference*.

Service endpoints for EBS direct APIs

An *endpoint* is a URL that serves as an entry point for an AWS web service. EBS direct APIs supports the following endpoint types:

- IPv4 endpoints
- Dual-stack endpoints that support both IPv4 and IPv6
- FIPS endpoints

When you make a request, you can specify the endpoint and Region to use. If you do not specify an endpoint, the IPv4 endpoint is used by default. To use a different endpoint type, you must specify it in your request. For examples of how to do this, see Specifying endpoints.

For more information about Regions, see Regions and Availability Zones in the *Amazon EC2 User Guide*. For a list of endpoints for EBS direct APIs, see Endpoints for the EBS direct APIs in the *Amazon Web Services General Reference*.

Topics

IPv4 endpoints

IPv4 endpoints support IPv4 traffic only. IPv4 endpoints are available for all Regions.

EBS direct APIs supports only Regional IPv4 endpoints that you can use to make your requests. You must specify the Region as part of the endpoint name. The endpoint names use the following naming convention:

- `ebs.region.amazonaws.com`

For example, to direct your requests to the `us-east-2` IPv4 endpoint, you must specify `ebs.us-east-2.amazonaws.com` as the endpoint. For a list of endpoints for EBS direct APIs, see [Endpoints for the EBS direct APIs](#) in the *Amazon Web Services General Reference*.

Pricing

You are not charged for data transferred directly between EBS direct APIs and Amazon EC2 instances using an IPv4 endpoint in the same Region. However, if there are intermediate services, such as AWS PrivateLink endpoints, NAT Gateway, or Amazon VPC Transit Gateways, you are charged their associated costs.

Dual-stack (IPv4 and IPv6) endpoints

Dual-stack endpoints support both IPv4 and IPv6 traffic. Dual-stack endpoints are available for all Regions.

To use IPv6, you must use a dual-stack endpoint. When you make a request to a dual-stack endpoint, the endpoint URL resolves to an IPv6 or an IPv4 address, depending on the protocol used by your network and client.

EBS direct APIs supports only regional dual-stack endpoints, which means that you must specify the Region as part of the endpoint name. Dual-stack endpoint names use the following naming convention:

- `ebs.region.api.aws`

For example, the dual-stack endpoint name for the `eu-west-1` Region is `ebs.eu-west-1.api.aws`. For a list of endpoints for EBS direct APIs, see [Endpoints for the EBS direct APIs](#) in the *Amazon Web Services General Reference*.

Pricing

You are not charged for data transferred directly between EBS direct APIs and Amazon EC2 instances using a dual-stack endpoint in the same Region. However, if there are intermediate services, such as AWS PrivateLink endpoints, NAT Gateway, or Amazon VPC Transit Gateways, you are charged their associated costs.

FIPS endpoints

EBS direct APIs provides FIPS-validated IPv4 and dual-stack (IPv4 and IPv6) endpoints for the following Regions:

- `us-east-1` — US East (N. Virginia)
- `us-east-2` — US East (Ohio)
- `us-west-1` — US West (N. California)
- `us-west-2` — US West (Oregon)
- `ca-central-1` — Canada (Central)
- `ca-west-1` — Canada West (Calgary)

FIPS IPv4 endpoints use the following naming convention: `ebs-fips.`*`region`*`.amazonaws.com`. For example, the FIPS IPv4 endpoint for `us-east-1` is `ebs-fips.us-east-1.amazonaws.com`.

FIPS dual-stack endpoints use the following naming convention: `ebs-fips.`*`region`*`.api.aws`. For example, the FIPS dual-stack endpoint for `us-east-1` is `ebs-fips.us-east-1.api.aws`.

For more information about FIPS endpoints see, FIPS endpoints in the *Amazon Web Services General Reference*.

Specifying endpoints

This section provides some examples of how to specify an endpoint when making a request.

AWS CLI

The following examples show how to specify an endpoint for the `us-east-2` Region using the AWS CLI.

- **Dual-stack**

```
aws ebs list-snapshot-blocks --snapshot-id snap-0987654321 --starting-block-index
 1000 --endpoint-url https://ebs.us-east-2.api.aws
```

- **IPv4**

```
aws ebs list-snapshot-blocks --snapshot-id snap-0987654321 --starting-block-index
 1000 --endpoint-url https://ebs.us-east-2.amazonaws.com
```

AWS SDK for Java 2.x

The following examples show how to specify an endpoint for the `us-east-2` Region using the AWS SDK for Java 2.x.

- **Dual-stack**

```
AwsClientBuilder.EndpointConfiguration config = new
 AwsClientBuilder.EndpointConfiguration("https://ebs.us-east-2.api.aws", "us-
east-2");
AmazonEBS ebs = AmazonEBSClientBuilder.standard()
    .withEndpointConfiguration(config)
    .build();
```

- **IPv4**

```
AwsClientBuilder.EndpointConfiguration config = new
 AwsClientBuilder.EndpointConfiguration("https://ebs.us-east-2.amazonaws.com",
 "us-east-2");
AmazonEBS ebs = AmazonEBSClientBuilder.standard()
    .withEndpointConfiguration(config)
    .build();
```

AWS SDK for Go

The following examples show how to specify an endpoint for the `us-east-2` Region using the AWS SDK for Go.

- **Dual-stack**

```
sess := session.Must(session.NewSession())
svc := ebs.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast2RegionID),
    Endpoint: aws.String("https://ebs.us-east-2.api.aws")
})
```

- **IPv4**

```
sess := session.Must(session.NewSession())
svc := ebs.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast2RegionID),
```

```
    Endpoint: aws.String("https://ebs.us-east-2.amazonaws.com")
})
```

AWS SDK code examples for EBS direct APIs

The following code examples show how to use EBS direct APIs with an AWS software development kit (SDK).

Actions

- Use StartSnapshot with an AWS SDK or CLI
- Use PutSnapshotBlock with an AWS SDK or CLI
- Use CompleteSnapshot with an AWS SDK or CLI

Use `StartSnapshot` with an AWS SDK or CLI

The following code example shows how to use `StartSnapshot`.

Rust

SDK for Rust

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
async fn start(client: &Client, description: &str) -> Result<String, Error> {
    let snapshot = client
        .start_snapshot()
        .description(description)
        .encrypted(false)
        .volume_size(1)
        .send()
        .await?;

    Ok(snapshot.snapshot_id.unwrap())
```

```
}
```

- For API details, see StartSnapshot in *AWS SDK for Rust API reference*.

Use PutSnapshotBlock with an AWS SDK or CLI

The following code example shows how to use `PutSnapshotBlock`.

Rust

SDK for Rust

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
async fn add_block(
    client: &Client,
    id: &str,
    idx: usize,
    block: Vec<u8>,
    checksum: &str,
) -> Result<(), Error> {
    client
        .put_snapshot_block()
        .snapshot_id(id)
        .block_index(idx as i32)
        .block_data(ByteStream::from(block))
        .checksum(checksum)
        .checksum_algorithm(ChecksumAlgorithm::ChecksumAlgorithmSha256)
        .data_length(EBS_BLOCK_SIZE as i32)
        .send()
        .await?;

    Ok(())
}
```

- For API details, see PutSnapshotBlock in *AWS SDK for Rust API reference*.

Use `CompleteSnapshot` with an AWS SDK or CLI

The following code example shows how to use `CompleteSnapshot`.

Rust

SDK for Rust

> **Note**
>
> There's more on GitHub. Find the complete example and learn how to set up and run in the AWS Code Examples Repository.

```
async fn finish(client: &Client, id: &str) -> Result<(), Error> {
    client
        .complete_snapshot()
        .changed_blocks_count(2)
        .snapshot_id(id)
        .send()
        .await?;

    println!("Snapshot ID {}", id);
    println!("The state is 'completed' when all of the modified blocks have been
 transferred to Amazon S3.");
    println!("Use the get-snapshot-state code example to get the state of the
 snapshot.");

    Ok(())
}
```

- For API details, see CompleteSnapshot in *AWS SDK for Rust API reference*.

Create a private connection between a VPC and EBS direct APIs

You can establish a private connection between your VPC and EBS direct APIs by creating an *interface VPC endpoint*, powered by AWS PrivateLink. You can access EBS direct APIs as if it were in your VPC, without using an internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with EBS direct APIs.

We create an endpoint network interface in each subnet that you enable for the interface endpoint.

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

Considerations for EBS direct APIs VPC endpoints

Before you set up an interface VPC endpoint for EBS direct APIs, review Considerations in the *AWS PrivateLink Guide*.

By default, full access to EBS direct APIs is allowed through the endpoint. You can control access to the interface endpoint using VPC endpoint policies. You can attach an endpoint policy to your VPC endpoint that controls access to EBS direct APIs. The policy specifies the following information:

- The **principal** that can perform actions.
- The **actions** that can be performed.
- The **resources** on which actions can be performed.

For more information, see Controlling access to services with VPC endpoints in the *Amazon VPC User Guide*.

The following is an example of an endpoint policy for EBS direct APIs. When attached to an endpoint, this policy grants access to all EBS direct APIs actions on all resources, except snapshots that are tagged with key `Environment` and value `Test`.

```
{
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ebs:*",
            "Principal": "*",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Environment": "Test"
```

```
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ebs:*",
            "Principal": "*",
            "Resource": "*"
        }
    ]
}
```

Create an interface VPC endpoint for EBS direct APIs

You can create a VPC endpoint for EBS direct APIs using either the Amazon VPC console or the AWS Command Line Interface (AWS CLI). For more information, see [Create a VPC endpoint] in the *AWS PrivateLink Guide*.

Create a VPC endpoint for EBS direct APIs using one of the following service names:

- `com.amazonaws.region.ebs`
- `com.amazonaws.region.ebs-fips` — To create an interface VPC endpoint that complies with the Federal Information Processing Standard (FIPS) Publication 140-2 US government standard.

> **Note**
>
> FIPS-compliant interface VPC endpoints can be created for the following Regions: `us-east-1` | `us-east-2` | `us-west-1` | `us-west-2` | `ca-central-1` | `ca-west-1`. FIPS-compliant interface VPC endpoints support both IPv4 and IPv6 traffic.

If you enable private DNS for the endpoint, you can make API requests to EBS direct APIs using its default DNS name for the Region, for example, `ebs.us-east-1.amazonaws.com`.

Log EBS direct APIs calls using AWS CloudTrail

EBS direct APIs are integrated with AWS CloudTrail, a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures calls made to the EBS direct APIs as events. The calls captured include calls from the AWS Management Console and code calls to the

EBS direct APIs. Using the information collected by CloudTrail, you can determine the request that was made to the EBS direct APIs, the IP address from which the request was made, when it was made, and additional details.

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root user or user credentials.
- Whether the request was made on behalf of an IAM Identity Center user.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

CloudTrail is active in your AWS account when you create the account and you automatically have access to the CloudTrail **Event history**. The CloudTrail **Event history** provides a viewable, searchable, downloadable, and immutable record of the past 90 days of recorded management events in an AWS Region. For more information, see Working with CloudTrail Event history in the *AWS CloudTrail User Guide*. There are no CloudTrail charges for viewing the **Event history**.

For an ongoing record of events in your AWS account past 90 days, create a trail or a CloudTrail Lake event data store.

CloudTrail trails

A *trail* enables CloudTrail to deliver log files to an Amazon S3 bucket. All trails created using the AWS Management Console are multi-Region. You can create a single-Region or a multi-Region trail by using the AWS CLI. Creating a multi-Region trail is recommended because you capture activity in all AWS Regions in your account. If you create a single-Region trail, you can view only the events logged in the trail's AWS Region. For more information about trails, see Creating a trail for your AWS account and Creating a trail for an organization in the *AWS CloudTrail User Guide*.

You can deliver one copy of your ongoing management events to your Amazon S3 bucket at no charge from CloudTrail by creating a trail, however, there are Amazon S3 storage charges. For more information about CloudTrail pricing, see AWS CloudTrail Pricing. For information about Amazon S3 pricing, see Amazon S3 Pricing.

CloudTrail Lake event data stores

CloudTrail Lake lets you run SQL-based queries on your events. CloudTrail Lake converts existing events in row-based JSON format to Apache ORC format. ORC is a columnar storage format that is optimized for fast retrieval of data. Events are aggregated into *event data stores*, which are immutable collections of events based on criteria that you select by applying advanced event selectors. The selectors that you apply to an event data store control which events persist and are available for you to query. For more information about CloudTrail Lake, see Working with AWS CloudTrail Lake in the *AWS CloudTrail User Guide*.

CloudTrail Lake event data stores and queries incur costs. When you create an event data store, you choose the pricing option you want to use for the event data store. The pricing option determines the cost for ingesting and storing events, and the default and maximum retention period for the event data store. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

EBS direct APIs data events in CloudTrail

Data events provide information about the resource operations performed on or in a resource. These are also known as data plane operations. Data events are often high-volume activities. By default, CloudTrail doesn’t log data events. The CloudTrail **Event history** doesn't record data events.

Additional charges apply for data events. For more information about CloudTrail pricing, see AWS CloudTrail Pricing.

You can log data events for the EBS direct APIs resource types by using the CloudTrail console, AWS CLI, or CloudTrail API operations. For more information about how to log data events, see Logging data events with the AWS Management Console and Logging data events with the AWS Command Line Interface in the *AWS CloudTrail User Guide*.

You can log the following EBS direct APIs operations as data events.

- ListSnapshotBlocks
- ListChangedBlocks
- GetSnapshotBlock
- PutSnapshotBlock

> **Note**
>
> If you perform an action on a snapshot that is shared with you, data events are not sent to the AWS account that owns the snapshot.

EBS direct APIs management events in CloudTrail

Management events provide information about management operations that are performed on resources in your AWS account. These are also known as control plane operations. By default, CloudTrail logs management events.

The EBS direct APIs service logs the following control plane operations to CloudTrail as management events.

- StartSnapshot
- CompleteSnapshot

EBS direct APIs event examples

An event represents a single request from any source and includes information about the requested API operation, the date and time of the operation, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so events don't appear in any specific order.

The following are example CloudTrail events for the EBS direct APIs.

StartSnapshot


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
```

```
    "eventTime": "2020-07-03T23:27:26Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "StartSnapshot",
    "awsRegion": "eu-west-1",
    "sourceIPAddress": "192.0.2.0",
    "userAgent": "PostmanRuntime/7.25.0",
    "requestParameters": {
        "volumeSize": 8,
        "clientToken": "token",
        "encrypted": true
    },
    "responseElements": {
        "snapshotId": "snap-123456789012",
        "ownerId": "123456789012",
        "status": "pending",
        "startTime": "Jul 3, 2020 11:27:26 PM",
        "volumeSize": 8,
        "blockSize": 524288,
        "kmsKeyArn": "HIDDEN_DUE_TO_SECURITY_REASONS"
    },
    "requestID": "be112233-1ba5-4ae0-8e2b-1c302EXAMPLE",
    "eventID": "6e12345-2a4e-417c-aa78-7594fEXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


CompleteSnapshot


```
{
    "eventVersion": "1.05",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "123456789012",
        "arn": "arn:aws:iam::123456789012:root",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
    "eventTime": "2020-07-03T23:28:24Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "CompleteSnapshot",
    "awsRegion": "eu-west-1",
    "sourceIPAddress": "192.0.2.0",
```

```
    "userAgent": "PostmanRuntime/7.25.0",
    "requestParameters": {
        "snapshotId": "snap-123456789012",
        "changedBlocksCount": 5
    },
    "responseElements": {
        "status": "completed"
    },
    "requestID": "be112233-1ba5-4ae0-8e2b-1c302EXAMPLE",
    "eventID": "6e12345-2a4e-417c-aa78-7594fEXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```


ListSnapshotBlocks


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAT4HPB2AO3JEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/user",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
    "eventTime": "2021-06-03T00:32:46Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "ListSnapshotBlocks",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "111.111.111.111",
    "userAgent": "PostmanRuntime/7.28.0",
    "requestParameters": {
        "snapshotId": "snap-abcdef01234567890",
        "maxResults": 100,
        "startingBlockIndex": 0
    },
    "responseElements": null,
    "requestID": "example6-0e12-4aa9-b923-1555eexample",
    "eventID": "example4-218b-4f69-a9e0-2357dexample",
    "readOnly": true,
    "resources": [
        {
```

```
            "accountId": "123456789012",
            "type": "AWS::EC2::Snapshot",
            "ARN": "arn:aws:ec2:us-west-2::snapshot/snap-abcdef01234567890"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-SHA",
        "clientProvidedHostHeader": "ebs.us-west-2.amazonaws.com"
    }
}
```


ListChangedBlocks


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAT4HPB2AO3JEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/user",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
    "eventTime": "2021-06-02T21:11:46Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "ListChangedBlocks",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "111.111.111.111",
    "userAgent": "PostmanRuntime/7.28.0",
    "requestParameters": {
        "firstSnapshotId": "snap-abcdef01234567890",
        "secondSnapshotId": "snap-9876543210abcdef0",
        "maxResults": 100,
        "startingBlockIndex": 0
    },
    "responseElements": null,
    "requestID": "example0-f4cb-4d64-8d84-72e1bexample",
    "eventID": "example3-fac4-4a78-8ebb-3e9d3example",
```

```
    "readOnly": true,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::EC2::Snapshot",
            "ARN": "arn:aws:ec2:us-west-2::snapshot/snap-abcdef01234567890"
        },
       {
            "accountId": "123456789012",
            "type": "AWS::EC2::Snapshot",
            "ARN": "arn:aws:ec2:us-west-2::snapshot/snap-9876543210abcdef0"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-SHA",
        "clientProvidedHostHeader": "ebs.us-west-2.amazonaws.com"
    }
}
```


GetSnapshotBlock


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAT4HPB2AO3JEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/user",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
    "eventTime": "2021-06-02T20:43:05Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "GetSnapshotBlock",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "111.111.111.111",
    "userAgent": "PostmanRuntime/7.28.0",
    "requestParameters": {
```

```
        "snapshotId": "snap-abcdef01234567890",
        "blockIndex": 1,
        "blockToken": "EXAMPLEiL5E3pMPFpaDWjExM2/mnSKh1mQfcbjwe2mM7EwhrgCdPAEXAMPLE"
    },
    "responseElements": null,
    "requestID": "examplea-6eca-4964-abfd-fd9f0example",
    "eventID": "example6-4048-4365-a275-42e94example",
    "readOnly": true,
    "resources": [
        {
          "accountId": "123456789012",
          "type": "AWS::EC2::Snapshot",
          "ARN": "arn:aws:ec2:us-west-2::snapshot/snap-abcdef01234567890"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-SHA",
        "clientProvidedHostHeader": "ebs.us-west-2.amazonaws.com"
    }
}
```


PutSnapshotBlock


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAT4HPB2AO3JEXAMPLE",
        "arn": "arn:aws:iam::123456789012:user/user",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "userName": "user"
    },
    "eventTime": "2021-06-02T21:09:17Z",
    "eventSource": "ebs.amazonaws.com",
    "eventName": "PutSnapshotBlock",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "111.111.111.111",
```

```
    "userAgent": "PostmanRuntime/7.28.0",
    "requestParameters": {
        "snapshotId": "snap-abcdef01234567890",
        "blockIndex": 1,
        "dataLength": 524288,
        "checksum": "exampleodSGvFSb1e3kxWUgbOQ4TbzPurnsfVexample",
        "checksumAlgorithm": "SHA256"
    },
    "responseElements": {
        "checksum": "exampleodSGvFSb1e3kxWUgbOQ4TbzPurnsfVexample",
        "checksumAlgorithm": "SHA256"
    },
    "requestID": "example3-d5e0-4167-8ee8-50845example",
    "eventID": "example8-4d9a-4aad-b71d-bb31fexample",
    "readOnly": false,
    "resources": [
        {
            "accountId": "123456789012",
            "type": "AWS::EC2::Snapshot",
            "ARN": "arn:aws:ec2:us-west-2::snapshot/snap-abcdef01234567890"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": false,
    "recipientAccountId": "123456789012",
    "eventCategory": "Data",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-SHA",
        "clientProvidedHostHeader": "ebs.us-west-2.amazonaws.com"
    }
}
```


For information about CloudTrail record contents, see CloudTrail record contents in the *AWS CloudTrail User Guide*.

Frequently asked questions for EBS direct APIs

Can a snapshot be accessed using the EBS direct APIs if it has a pending status?

No. The snapshot can be accessed only if it has a completed status.

Are the block indexes returned by the EBS direct APIs in numerical order?

Yes. The block indexes returned are unique, and in numerical order.

Can I submit a request with a MaxResults parameter value of under 100?

No. The minimum MaxResult parameter value you can use is 100. If you submit a request with a MaxResult parameter value of under 100, and there are more than 100 blocks in the snapshot, then the API will return at least 100 results.

Can I run API requests concurrently?

You can run API requests concurrently. Be sure to take note of other workloads that might be running in the account to avoid bottlenecks. You should also build retry mechanisms into your EBS direct APIs workflows to handle throttling, timeouts, and service unavailability. For more information, see Optimize performance for EBS direct APIs.

Review the EBS direct APIs service quotas to determine the API requests that you can run per second. For more information, see Amazon Elastic Block Store Endpoints and Quotas in the *AWS General Reference*.

When running the ListChangedBlocks action, is it possible to get an empty response even though there are blocks in the snapshot?

Yes. If the changed blocks are scarce in the snapshot, the response may be empty but the API will return a next page token value. Use the next page token value to continue to the next page of results. You can confirm that you have reached the last page of results when the API returns a next page token value of null.

If the NextToken parameter is specified together with a StartingBlockIndex parameter, which of the two is used?

The NextToken is used, and the StartingBlockIndex is ignored.

How long are the block tokens and next tokens valid?

Block tokens are valid for seven days, and next tokens are valid for 60 minutes.

Are encrypted snapshots supported?

Yes. Encrypted snapshots can be accessed using the EBS direct APIs.

To access an encrypted snapshot, the user must have access to the KMS key used to encrypt the snapshot, and the AWS KMS decrypt action. See the Control access to EBS direct APIs using IAM section earlier in this guide for the AWS KMS policy to assign to a user.

Are public snapshots supported?

Public snapshots are not supported.

Are Amazon EBS local snapshots on AWS Outposts supported?

Amazon EBS local snapshots on AWS Outposts are not supported.

Does list snapshot block return all block indexes and block tokens in a snapshot, or only those that have data written to them?

It returns only block indexes and tokens that have data written to them.

Can I get a history of the API calls made by the EBS direct APIs on my account for security analysis and operational troubleshooting purposes?

Yes. To receive a history of EBS direct APIs API calls made on your account, turn on AWS CloudTrail in the AWS Management Console. For more information, see Log EBS direct APIs calls using AWS CloudTrail.

Recover deleted EBS volumes, EBS snapshots, and EBS-backed AMIs with Recycle Bin

Recycle Bin is a data recovery feature that enables you to restore accidentally deleted EBS volumes, EBS snapshots, and EBS-backed AMIs. When using Recycle Bin, if your resources are deleted, they are retained in the Recycle Bin for a time period that you specify before being permanently deleted.

You can restore a resource from the Recycle Bin at any time before its retention period expires. After you restore a resource from the Recycle Bin, the resource is removed from the Recycle Bin and you can use it in the same way that you use any other resource of that type in your account. If the retention period expires and the resource is not restored, the resource is permanently deleted from the Recycle Bin and it is no longer available for recovery.

Using Recycle Bin helps to ensure business continuity by protecting your business-critical data against accidental deletion.

Recycle Bin is assessed as a service capability of Amazon Elastic Block Store (Amazon EBS). Any AWS services in Scope by Compliance Program (FedRAMP, HIPAA BAA, SOC, etc) which lists Amazon EBS will also apply to Recycle Bin.

Topics

- Delete a Recycle Bin retention rule to stop it from retaining resources
- Recover deleted snapshots from the Recycle Bin
- Recover deleted volumes from the Recycle Bin
- Recover deleted AMIs from the Recycle Bin
- Monitor Recycle Bin using Amazon EventBridge
- Monitor Recycle Bin using AWS CloudTrail
- Service endpoints for Recycle Bin
- Create a private connection between a VPC and Recycle Bin

Supported resources

Recycle Bin supports the following resource types:

- Amazon EBS volumes
- Amazon EBS snapshots

> **Important**
>
> Recycle Bin retention rules also apply to archived snapshots in the archive storage tier. If you delete an archived snapshot that matches a retention rule, that snapshot is retained in the Recycle Bin for the period defined in the retention rule. Archived snapshots are billed at the rate for archived snapshots while they are in the Recycle Bin.

- Amazon EBS-backed Amazon Machine Images (AMIs)

> **Note**
>
> Retention rules also apply to disabled AMIs.

How does Recycle Bin work?

To enable and use Recycle Bin, you must create *retention rules* in the AWS Regions in which you want to protect your resources. Retention rules specify the following:

- The resource type that you want to protect (volumes, snapshots, or AMIs).

- The type of retention rule:
  - **Tag-level retention rules** — These retention rules use resource tags to identify the resources to protect. For each retention rule, you specify one or more tag key and value pairs. Resources (of the specified type) that have at least one of these tag key and value pairs are automatically retained in the Recycle Bin upon deletion. Use this type of retention rule to protect specific resources in your account based on their tags.
  - **Region-level retention rules** — These retention rules, by default, apply to all of the resources (of the specified type) in the Region, even if the resources are not tagged. However, you can specify exclusion tags to exclude resources that have specific tags. Use this type of retention rule to protect all resources of a specific type in a Region.
- The retention period to retain resources after they are deleted. After this period expires, the resources are permanently deleted from the Recycle Bin. The supported retention periods are:
  - EBS volumes: 1 - 7 days
  - EBS snapshots and EBS-backed AMIs: 1 - 365 days

While a resource is in the Recycle Bin, you have the ability to restore it for use at any time. The resource remains in the Recycle Bin until one of the following happens:

- You manually restore it for use. When you restore a resource from the Recycle Bin, the resource is removed from the Recycle Bin and it immediately becomes available for use. You can use restored resources in the same way as any other resource of that type in your account.
- The retention period expires. If the retention period expires, and the resource has not been restored from the Recycle Bin, the resource is permanently deleted from the Recycle Bin and it can no longer be viewed or restored.

Considerations for Recycle Bin

The following considerations apply when working with Recycle Bin and retention rules.

General considerations

- Deleted resources are moved to the Recycle Bin only if they match an existing retention rule. If you delete a resource that does not match a retention rule, or if you do not have any retention rules at that time, that resource is permanently deleted; it is not moved to the Recycle Bin.

- > **Important**
  >
  > Retention rules follow an eventual consistency model for the first retention rule created per resource type, per Region in your account. When you create your first retention rule for a resource type in a Region, that rule might not become active and start retaining resources immediately. However, any subsequent retention rules you create for that same resource type in the same Region will become active and start retaining resources almost immediately.

- If a resource matches more than one retention rule upon deletion, then the retention rule with the longest retention period takes precedence.
- You can't manually delete a resource from the Recycle Bin. The resource will be automatically deleted when its retention period expires.
- While a resource is in the Recycle Bin, you can only view it, restore it, or modify its tags. To use the resource in any other way, you must first restore it.
- If any AWS service, such as AWS Backup or Amazon Data Lifecycle Manager, deletes a resource that matches a retention rule, that resource is automatically retained by Recycle Bin. If needed, you can prevent these resources from entering into Recycle Bin upon deletion by tagging those resources and then adding those tags as exclusion tags to your retention rules.
- When a resource is sent to the Recycle Bin, the following system-generate tag is assigned to the resource:
  - Tag key — `aws:recycle-bin:resource-in-bin`
  - Tag value — `true`

  You can't manually edit or delete this tag. When the resource is restored from the Recycle Bin, the tag is automatically removed.

Considerations for volumes

- Volumes deleted due to instance termination or root volume replacement are protected by Recycle Bin.
- Volumes that fail to be created are not protected by Recycle Bin on deletion.
- Volumes of failed instance launches are not protected by Recycle Bin on deletion.
- Volumes of managed instances are not protected by Recycle Bin on deletion.

- Ongoing volume creation or modification will not be paused when the volume enters Recycle Bin. This means that you are still billed accordingly if the volume was created with an Amazon EBS Provisioned Rate for Volume Initialization.
- Volumes in Recycle Bin count towards your quotas in the same way as regular volumes.
- Volumes in Recycle Bin are not billed after their Recycle Bin exit time has elapsed. You cannot restore these volumes but you can discover them if they have not yet been deleted.
- The `deleteVolume` event will be sent only after the volume is deleted from Recycle Bin. This event is not emitted when the volume enters Recycle Bin.

Considerations for snapshots

- **Important**

  If you have retention rules for AMIs and for their associated snapshots, make the retention period for the snapshots the same or longer than the retention period for the AMIs. This ensures that Recycle Bin does not delete the snapshots associated with an AMI before deleting the AMI itself, as this would make the AMI unrecoverable.
- If a snapshot is enabled for fast snapshot restore when it is deleted, fast snapshot restore is automatically disabled shortly after the snapshot is sent to the Recycle Bin.
  - If you restore the snapshot before fast snapshot restore is disabled for the snapshot, it remains enabled.
  - If you restore the snapshot, after fast snapshot restore has been disabled, it remains disabled. If needed, you must manually re-enable fast snapshot restore.
- If a snapshot is shared when it is deleted, it is automatically unshared when it is sent to the Recycle Bin. If you restore the snapshot, all of the previous sharing permissions are automatically restored.
- If a snapshot that was created by another AWS service, such as AWS Backup is sent to the Recycle Bin and you later restore that snapshot from the Recycle Bin, it is no longer managed by the AWS service that created it. You must manually delete the snapshot if it is no longer needed.

Considerations for AMIs

- Only Amazon EBS-backed AMIs are supported.

- > **Important**
  >
  > If you have retention rules for AMIs and for their associated snapshots, make the retention period for the snapshots the same or longer than the retention period for the AMIs. This ensures that Recycle Bin does not delete the snapshots associated with an AMI before deleting the AMI itself, as this would make the AMI unrecoverable.

- If an AMI is shared when it is deleted, it is automatically unshared when it is sent to the Recycle Bin. If you restore the AMI, all of the previous sharing permissions are automatically restored.
- Before you can restore an AMI from the Recycle Bin, you must first restore all of its associated snapshots from the Recycle Bin and ensure that they are in the `available` state.
- If the snapshots that are associated with the AMI are deleted from the Recycle Bin, the AMI is no longer recoverable. The AMI will be deleted when the retention period expires.
- If an AMI that was created by another AWS service, such as AWS Backup, is sent to the Recycle Bin and you later restore that AMI from the Recycle Bin, it is no longer managed by the AWS service that created it. You must manually delete the AMI if it is no longer needed.

Considerations for Amazon Data Lifecycle Manager snapshot policies

- If Amazon Data Lifecycle Manager deletes a snapshot that matches a retention rule, that snapshot is automatically retained by Recycle Bin.
- If Amazon Data Lifecycle Manager deletes a snapshot and sends it to the Recycle Bin when the policy's retention threshold is reached, and you manually restore the snapshot from the Recycle Bin, you must manually delete that snapshot when it is no longer needed. Amazon Data Lifecycle Manager will no longer manage the snapshot.
- If you manually delete a snapshot that was created by a policy, and that snapshot is in the Recycle Bin when the policy’s retention threshold is reached, Amazon Data Lifecycle Manager will not delete the snapshot. Amazon Data Lifecycle Manager does not manage the snapshots while they are stored in the Recycle Bin.

  If the snapshot is restored from the Recycle Bin before the policy's retention threshold is reached, Amazon Data Lifecycle Manager will delete the snapshot when the policy's retention threshold is reached.

If the snapshot is restored from the Recycle Bin after the policy's retention threshold is reached, Amazon Data Lifecycle Manager will no longer delete the snapshot. You must manually delete the snapshot when it is no longer needed.

Considerations for AWS Backup

- If AWS Backup deletes a snapshot that matches a retention rule, that snapshot is automatically retained by Recycle Bin.

Considerations for archived snapshots

- Recycle Bin retention rules also apply to archived snapshots in the archive storage tier. If you delete an archived snapshot that matches a retention rule, that snapshot is retained in the Recycle Bin for the period defined in the retention rule.

  Archived snapshots are billed at the rate for archived snapshots while they are in the Recycle Bin.

  If a retention rule deletes an archived snapshot from the Recycle Bin before the minimum archive period of 90 days, you are billed for the remaining days. For more information, see Archived snapshot pricing and billing.

  To use an archived snapshot that is in the Recycle Bin, you must first recover the snapshot from the Recycle Bin and then restore it from the archive tier to the standard tier.

Quotas

The following quotas apply to Recycle Bin.

Quota	Default quota
Retention rules per Region	250
Tag key and value pairs	50

Quota	Default quota			
per retention rule				

Related services

Recycle Bin works with the following services:

- **AWS CloudTrail** — Enables you to record events that occur in Recycle Bin. For more information, see Monitor Recycle Bin using AWS CloudTrail.

Pricing

There are no additional charges for using Recycle Bin and retention rules. For more information, see Amazon EBS pricing.

- **Amazon EBS volumes** — Volumes in the Recycle Bin are billed at the same rate as regular volumes in your account.
- **Amazon EBS snapshots** — Snapshots in the Recycle Bin are billed at the same rate as regular snapshots in your account.
- **EBS-backed AMIs** — AMIs in the Recycle Bin do not incur any additional charges.

> **Note**
>
> Some resources might still appear in the Recycle Bin console or in the AWS CLI and API output for a short period after their retention periods have expired and they have been permanently deleted. You are not billed for these resources. Billing stops as soon as the retention period expires.

You can use the following AWS generated cost allocation tags for cost tracking and allocation purposes when using AWS Billing and Cost Management.

- Key: `aws:recycle-bin:resource-in-bin`
- Value: `true`

For more information, see AWS-generated cost allocation tags in the *AWS Billing and Cost Management User Guide*.

Control access to Recycle Bin with IAM

By default, users don't have permission to work with Recycle Bin, retention rules, or with resources that are in the Recycle Bin. To allow users to work with these resources, you must create IAM policies that grant permission to use specific resources and API actions. After the policies are created, you must add permissions to your users, groups, or roles.

Topics

Permissions for working with Recycle Bin and retention rules

To work with Recycle Bin and retention rules, users need the following permissions.

- `rbin:CreateRule`
- `rbin:UpdateRule`
- `rbin:GetRule`
- `rbin:ListRules`
- `rbin:DeleteRule`
- `rbin:TagResource`
- `rbin:UntagResource`
- `rbin:ListTagsForResource`
- `rbin:LockRule`
- `rbin:UnlockRule`

To use the Recycle Bin console, users need the `tag:GetResources` permission.

The following is an example IAM policy that includes the `tag:GetResources` permission for console users. If some permissions are not needed, you can remove them from the policy.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "rbin:CreateRule",
                "rbin:UpdateRule",
                "rbin:GetRule",
                "rbin:ListRules",
                "rbin:DeleteRule",
                "rbin:TagResource",
                "rbin:UntagResource",
                "rbin:ListTagsForResource",
                "rbin:LockRule",
                "rbin:UnlockRule",
                "tag:GetResources"
            ],
            "Resource": "*"
        }
    ]
}
```

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.

- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.

- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.

- (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

Permissions for working with resources in the Recycle Bin

For more information about the IAM permissions needed to work with resources in the Recycle Bin, see the following:

- Permissions for working with volumes in the Recycle Bin
- Permissions for working with snapshots in the Recycle Bin
- Permissions for working with AMIs in the Recycle Bin

Condition keys for Recycle Bin

Recycle Bin defines the following condition keys that you can use in the `Condition` element of an IAM policy to control the conditions under which the policy statement applies. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.

Topics

- rbin:Request/ResourceType condition key
- rbin:Attribute/ResourceType condition key

`rbin:Request/ResourceType` condition key

The `rbin:Request/ResourceType` condition key can be used to filter access on CreateRule and ListRules requests based on the value specified for the `ResourceType` request parameter.

Example 1 - CreateRule

The following sample IAM policy allows IAM principals to make **CreateRule** requests only if the value specified for the `ResourceType` request parameter is `EBS_SNAPSHOT` or `EC2_IMAGE`. This allows the principal to create new retention rules for snapshots and AMIs only.

JSON

```
{
```

```
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:CreateRule"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Request/ResourceType" : ["EBS_SNAPSHOT", "EC2_IMAGE"]
                }
            }
        }
    ]
}
```

Example 2 - ListRules

The following sample IAM policy allows IAM principals to make **ListRules** requests only if the value specified for the `ResourceType` request parameter is `EBS_SNAPSHOT`. This allows the principal to list retention rules for snapshots only, and it prevents them from listing retention rules for any other resource type.

JSON

```
{
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:ListRules"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Request/ResourceType" : "EBS_SNAPSHOT"
                }
            }
        }
```

```
    ]
}
```

`rbin:Attribute/ResourceType` condition key

The `rbin:Attribute/ResourceType` condition key can be used to filter access on DeleteRule, GetRule, UpdateRule, LockRule, UnlockRule, TagResource, UntagResource, and ListTagsForResource requests based on the value of the retention rule's `ResourceType` attribute.

Example 1 - UpdateRule

The following sample IAM policy allows IAM principals to make **UpdateRule** requests only if the `ResourceType` attribute of the requested retention rule is `EBS_SNAPSHOT` or `EC2_IMAGE`. This allows the principal to update retention rules for snapshots and AMIs only.

JSON

```
{
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:UpdateRule"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Attribute/ResourceType" : ["EBS_SNAPSHOT", "EC2_IMAGE"]
                }
            }
        }
    ]
}
```

Example 2 - DeleteRule

The following sample IAM policy allows IAM principals to make **DeleteRule** requests only if the `ResourceType` attribute of the requested retention rule is `EBS_SNAPSHOT`. This allows the principal to delete retention rules for snapshots only.

JSON

```
{
    "Version":"2012-10-17",
    "Statement" : [
        {
            "Effect" : "Allow",
            "Action" :[
                "rbin:DeleteRule"
            ],
            "Resource" : "*",
            "Condition" : {
                "StringEquals" : {
                    "rbin:Attribute/ResourceType" : "EBS_SNAPSHOT"
                }
            }
        }
    ]
}
```

Create a Recycle Bin retention rule

When you create a retention rule, you must specify the following required parameters:

- The resource type to protect (volumes, snapshots, or AMIs).
- The type of retention rule (tag-level or Region-level). Tag-level rules protect only resources that have specific tags. Region-level rules protect all resources in the Region, but can exclude resources that have specific tags.
- The retention period to retain resources after they are deleted. After this period expires, the resources are permanently deleted from the Recycle Bin. The supported retention periods are:
  - EBS volumes: 1 - 7 days
  - EBS snapshots and EBS-backed AMIs: 1 - 365 days

You can also optionally specify a rule name and description of up to 255 characters each, and tags to help you identify and organize your rules. We recommend that you do not include personally identifying, confidential, or sensitive information in the name, description, or tags.

You can also optionally lock Region-level retention rules on creation. If you lock a retention rule on creation, you must also specify the unlock delay period, which can be 7 to 30 days. Retention rules remain unlocked by default unless you explicitly lock them.

> **Note**
>
> Retention rules function only in the Regions in which they are created. If you intend to use Recycle Bin in other Regions, you must create additional retention rules in those Regions.

You can create a Recycle Bin retention rule using one of the following methods.

Recycle Bin console

To create a tag-level retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**, and then choose **Create retention rule**.
3. (*Optional*) For **Retention rule name**, enter a descriptive name for the retention rule.
4. (*Optional*) For **Retention rule description**, enter a brief description for the retention rule.
5. For **Resource type**, select the type of resource for the retention rule to protect. The retention rule will retain only resources of this type in the Recycle Bin.
6. For **Select the resources to retain**, choose **Retain resources that have specific tags**.
7. For **Resource tags**, enter the tag key and value pairs to use to identify the resources to retain in the Recycle Bin. Only resources of the specified type that have at least one of the specified tag will be retained by the retention rule.
8. For **Retention period**, enter the number of days to retain deleted resources in the Recycle Bin.
9. Choose **Create retention rule**.

To create a Region-level retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/

2. In the navigation pane, choose **Retention rules**, and then choose **Create retention rule**.
3. (*Optional*) For **Retention rule name**, enter a descriptive name for the retention rule.
4. (*Optional*) For **Retention rule description**, enter a brief description for the retention rule.
5. For **Resource type**, select the type of resource for the retention rule to protect. The retention rule will retain only resources of this type in the Recycle Bin.
6. For **Select the resources to retain**, choose **Retain all resources**.
7. (*Optional*) To exclude resources that have specific tags, for **Exclusion tags**, enter up to five tag key and value pairs to use to identify the resources to exclude. Resources that have any of these tags are ignored by the retention rule.
8. For **Retention period**, enter the number of days to retain deleted resources in the Recycle Bin.
9. (*Optional*) To lock the retention rule, for **Rule lock settings**, select **Lock**, and then for **Unlock delay period**, specify the unlock delay period in days. A locked retention rule can't be modified or deleted. To modify or delete the rule, you must first unlock it and then wait for the unlock delay period to expire. For more information, see [Lock a Recycle Bin retention rule to prevent it from being updated or deleted](#)

   To leave the retention rule unlocked, for **Rule lock settings**, keep **Unlock** selected. An unlocked retention rule can be modified or deleted at any time.

   > **Note**
   >
   > You can't lock Region-level retention rules that have exclusion tags.

10. Choose **Create retention rule**.

AWS CLI

To create a retention rule

Use the [create-rule](#) AWS CLI command. For `--retention-period`, specify the number of days to retain deleted snapshots in the Recycle Bin. For `--resource-type`, specify `EBS_VOLUME` for volumes, `EBS_SNAPSHOT` for snapshots, or `EC2_IMAGE` for AMIs. To create a tag-level retention rule, for `--resource-tags`, specify the tags to use to identify the resources that are to be retained. To create a Region-level retention rule, omit `--resource-tags`, and optionally specify `--exclude-resource-tags`, to exclude resources that have specific tags. To lock a

Region-level retention rule, include `--lock-configuration`, and specify the unlock delay period in days.

```
aws rbin create-rule \
--retention-period RetentionPeriodValue=number_of_days,RetentionPeriodUnit=DAYS \
--resource-type EBS_VOLUME|EBS_SNAPSHOT|EC2_IMAGE \
--description "rule_description" \
--lock-configuration
 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=unlock_delay_in_days}' \
--resource-tags ResourceTagKey=tag_key,ResourceTagValue=tag_value \
--exclude-resource-tags ResourceTagKey=tag_key,ResourceTagValue=tag_value
```

Example 1

The following example command creates an unlocked Region-level retention rule that retains all deleted snapshots for a period of 7 days.

```
aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match all snapshots"
```

Example 2

The following example command creates a tag-level rule that retains deleted snapshots that are tagged with `purpose=production` for a period of 7 days.

```
aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match snapshots with a specific tag" \
--resource-tags ResourceTagKey=purpose,ResourceTagValue=production
```

Example 3

The following example command creates a locked Region-level retention rule that retains all deleted snapshots for a period of 7 days. The retention rule is locked with an unlock delay period of 7 days.

```
aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
```

```
--description "Match all snapshots" \
--lock-configuration 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=7}'
```

Example 4

The following example command creates an unlocked Region-level retention rule that retains all deleted snapshots, except snapshots that are tagged with `purpose:testing`, for a period of 7 days.

```
aws rbin create-rule \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
--description "Match only production snapshots" \
--exclude-resource-tags ResourceTagKey=purpose,ResourceTagValue=testing
```

Update an existing Recycle Bin retention rule

You can update an unlocked retention rule's description, resource tags, and retention period at any time after creation. You can't update a retention rule's resource type or unlock delay period, even if the retention rule is unlocked.

You can't update a locked retention rule in any way. If you need to modify a locked retention rule, you must first unlock it and wait for the unlock delay period to expire.

If you need to modify the unlock delay period for a locked retention rule, you must unlock the retention rule, and wait for the current unlock delay period to expire. When the unlock delay period is expired, you must relock the retention rule and specify the new unlock delay period.

> **Note**
>
> We recommend that you do not include personally identifying, confidential, or sensitive information in the retention rule description.

After you update a retention rule, the changes only apply to new resources that it retains. The changes do not affect resources that it previously sent to the Recycle Bin. For example, if you update a retention rule's retention period, only snapshots that are deleted after the update are retained for the new retention period. Snapshots that it sent to the Recycle Bin before the update are still retained for the previous (old) retention period.

You can update a retention rule using one of the following methods.

Recycle Bin console

To update a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**.
3. In the grid, select the retention rule to update, and choose **Actions**, **Edit retention rule**.
4. In the **Rule details** section, update **Retention rule name** and **Retention rule description** as needed.
5. In the **Rule settings** section, update the **Resource type**, **Resource tags to match**, and **Retention period** as needed.
6. In the **Tags** section, add or remove retention rule tags as needed.
7. Choose **Save retention rule**.

AWS CLI

To update a retention rule

Use the update-rule AWS CLI command. For `--identifier`, specify the ID of the retention rule to update For `--resource-types`, specify `EBS_VOLUME` for volumes, `EBS_SNAPSHOT` for snapshots, or `EC2_IMAGE` for AMIs.

```
aws rbin update-rule \
--identifier rule_ID \
--retention-period RetentionPeriodValue=number_of_days,RetentionPeriodUnit=DAYS \
--resource-type EBS_VOLUME|EBS_SNAPSHOT|EC2_IMAGE \
--description "rule_description"
```

Example

The following example command updates retention rule `6lsJ2Fa9nh9` to retain all snapshots for 7 days and updates its description.

```
aws rbin update-rule \
--identifier 6lsJ2Fa9nh9 \
--retention-period RetentionPeriodValue=7,RetentionPeriodUnit=DAYS \
--resource-type EBS_SNAPSHOT \
```

```
--description "Retain for three weeks"
```

Lock a Recycle Bin retention rule to prevent it from being updated or deleted

Recycle Bin lets you lock Region-level retention rules at any time.

A locked retention rule can't be modified or deleted, even by users who have the required IAM permissions. Lock your retention rules to help protect them against accidental or malicious modifications and deletions.

When you lock a retention rule, you must specify an unlock delay period. This is the period of time that you must wait after unlocking the retention rule before you can modify or delete it. You cannot modify or delete the retention rule during the unlock delay period. You can modify or delete the retention rule only after the unlock delay period has expired.

You can't change the unlock delay period after the retention rule has been locked. If your account permissions have been compromised, the unlock delay period gives you additional time to detect and respond to security threats. The length of this period should be longer than the time it takes for you to identify and respond to security breaches. To set the right duration, you can review previous security incidents and the time needed to identify and remediate an account breach.

We recommend that you use Amazon EventBridge rules to notify you of retention rule lock state changes. For more information, see Monitor Recycle Bin using Amazon EventBridge.

Considerations

- You can't lock tag-level retention rules, or Region-level retention rules that have exclusion tags.
- You can lock an unlocked retention rule at any time.
- The unlock delay period must be 7 to 30 days.
- You can re-lock a retention rule during the unlock delay period. Relocking the retention rule resets the unlock delay period.

You can lock a Region-level retention rule using one of the following methods.

Recycle Bin console

To lock a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation panel, choose **Retention rules**.
3. In the grid, select the unlocked retention rule to lock, and choose **Actions**, **Edit retention rule lock**.
4. In the Edit retention rule lock screen, choose **Lock**, and then for **Unlock delay period**, specify the unlock delay period in days.
5. Select the **I acknowledge that locking the retention rule will prevent it from being modified or deleted** check box, and then choose **Save**.

AWS CLI

To lock an unlocked retention rule

Use the lock-rule AWS CLI command. For `--identifier`, specify the ID of the retention rule to lock. For `--lock-configuration`, specify the unlock delay period in days.

```
aws rbin lock-rule \
--identifier rule_ID \
--lock-configuration
 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=number_of_days}'
```

Example

The following example command locks retention rule `6lsJ2Fa9nh9` and sets the unlock delay period to 15 days.

```
aws rbin lock-rule \
--identifier 6lsJ2Fa9nh9 \
--lock-configuration 'UnlockDelay={UnlockDelayUnit=DAYS,UnlockDelayValue=15}'
```

Unlock a Recycle Bin retention rule to allow it to be updated or deleted

You can't modify or delete a locked retention rule. If you need to modify a locked retention rule, you must first unlock it. After you have unlocked the retention rule, you must wait for the unlock delay period to expire before you can modify or delete it. You can't modify or delete a retention rule during the unlock delay period.

An unlocked retention rule can be modified and deleted at any time by a user who has the required IAM permissions. Leaving your retention rules unlocked could expose them to accidental or malicious modifications and deletions.

Considerations

- You can re-lock a retention rule during the unlock delay period.
- You can re-lock a retention rule after the unlock delay period has expired.
- You can't bypass the unlock delay period.
- You can't change the unlock delay period after the initial lock.

We recommend that you use Amazon EventBridge rules to notify you of retention rule lock state changes. For more information, see Monitor Recycle Bin using Amazon EventBridge.

You can unlock a locked Region-level retention rule using one of the following methods.

Recycle Bin console

To unlock a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation panel, choose **Retention rules**.
3. In the grid, select the locked retention rule to unlock, and choose **Actions**, **Edit retention rule lock**.
4. On the Edit retention rule lock screen, choose **Unlock**, and then choose **Save**.

AWS CLI

To unlock a locked retention rule

Use the unlock-rule AWS CLI command. For `--identifier`, specify the ID of the retention rule to unlock.

```
aws rbin unlock-rule \
--identifier rule_ID
```

Example

The following example command unlocks retention rule `6lsJ2Fa9nh9`

```
aws rbin unlock-rule \
--identifier 6lsJ2Fa9nh9
```

Tag a Recycle Bin retention rule

You can assign custom tags to your retention rules to categorize them in different ways, for example, by purpose, owner, or environment. This helps you to efficiently find a specific retention rule based on the custom tags that you assigned.

You can assign a tag to a retention rule using one of the following methods.

Recycle Bin console

To tag a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**.
3. Select the retention rule to tag, choose the **Tags** tab, and then choose **Manage tags**.
4. Choose **Add tag**. For **Key**, enter the tag key. For **Value**, enter the tag value.
5. Chose **Save**.

AWS CLI

To tag a retention rule

Use the tag-resource AWS CLI command. For `--resource-arn`, specify the Amazon Resource Name (ARN) of the retention rule to tag, and for `--tags`, specify the tag key and value pair.

```
aws rbin tag-resource \
--resource-arn retention_rule_arn \
--tags key=tag_key,value=tag_value
```

Example

The following example command tags retention rule `arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3` with tag `purpose=production`.

```
aws rbin tag-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 \
--tags key=purpose,value=production
```

View retention rule tags

You can view the tags assigned to a retention rule using one of the following methods.

Recycle Bin console

To view tags for a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**.
3. Select the retention rule for which to view the tags, and choose the **Tags** tab.

AWS CLI

To view the tags assigned to a retention rule

Use the list-tags-for-resource AWS CLI command. For `--resource-arn`, specify the ARN of the retention rule.

```
aws rbin list-tags-for-resource \
--resource-arn retention_rule_arn
```

Example

The following example command lists the tags for retention rule `arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3`.

```
aws rbin list-tags-for-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3
```

Remove tags from retention rules

You can remove tags from a retention rule using one of the following methods.

Recycle Bin console

To remove a tag from a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**.
3. Select the retention rule from which to remove the tag, choose the **Tags** tab, and then choose **Manage tags**.
4. Choose **Remove** next to the tag to remove.
5. Chose **Save**.

AWS CLI

To remove a tag from a retention rule

Use the untag-resource AWS CLI command. For `--resource-arn`, specify the ARN of the retention rule. For `--tagkeys`, specify the tags keys of the tags to remove.

```
aws rbin untag-resource \
--resource-arn retention_rule_arn \
--tagkeys tag_key
```

Example

The following example command removes tags that have a tag key of `purpose` from retention rule `arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3`.

```
aws rbin untag-resource \
--resource-arn arn:aws:rbin:us-east-1:123456789012:rule/nOoSBBtItF3 \
--tagkeys purpose
```

Delete a Recycle Bin retention rule to stop it from retaining resources

You can delete a retention rule at any time. When you delete a retention rule, it no longer retains new resources in the Recycle Bin after they have been deleted. Resources that were sent to the Recycle Bin before the retention rule was deleted continue to be retained in the Recycle Bin according to the retention period defined in the retention rule. When the period expires, the resource is permanently deleted from the Recycle Bin.

You can delete a retention rule using one of the following methods.

Recycle Bin console

To delete a retention rule

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Retention rules**.
3. In the grid, select the retention rule to delete, and choose **Actions**, **Delete retention rule**.
4. When prompted, enter the confirmation message and choose **Delete retention rule**.

AWS CLI

To delete a retention rule

Use the delete-rule AWS CLI command. For `--identifier`, specify the ID of the retention rule to delete.

```
aws rbin delete-rule --identifier rule_ID
```

Example

The following example command deletes retention rule `6lsJ2Fa9nh9`.

```
aws rbin delete-rule --identifier 6lsJ2Fa9nh9
```

Recover deleted snapshots from the Recycle Bin

This topic explains how to recover Amazon EBS snapshots from the Recycle Bin.

Topics

- [Permissions for working with snapshots in the Recycle Bin]
- [View snapshots in the Recycle Bin]
- [Restore snapshots from the Recycle Bin]

Permissions for working with snapshots in the Recycle Bin

By default, users don't have permission to work with snapshots that are in the Recycle Bin. To allow users to work with these resources, you must create IAM policies that grant permission to use specific resources and API actions. After the policies are created, you must add permissions to your users, groups, or roles.

To view and recover snapshots that are in the Recycle Bin, users must have the following permissions:

- `ec2:ListSnapshotsInRecycleBin`
- `ec2:RestoreSnapshotFromRecycleBin`

To manage tags for snapshots in the Recycle Bin, users need the following additional permissions.

- `ec2:CreateTags`
- `ec2:DeleteTags`

To use the Recycle Bin console, users need the `ec2:DescribeTags` permission.

The following is an example IAM policy. It includes the `ec2:DescribeTags` permission for console users, and it includes the `ec2:CreateTags` and `ec2:DeleteTags` permissions for managing tags. If the permissions are not needed, you can remove them from the policy.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in [Create a permission set] in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.

- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

For more information about the permissions needed to use Recycle Bin, see Permissions for working with Recycle Bin and retention rules.

View snapshots in the Recycle Bin

While a snapshot is in the Recycle Bin, you can view limited information about it, including:

- The ID of the snapshot.
- The snapshot description.
- The ID of the volume from which the snapshot was created.
- The date and time when the snapshot was deleted and it entered Recycle Bin.
- The date and time when the retention period expires. The snapshot will be permanently deleted from the Recycle Bin at this time.

You can view the snapshots in the Recycle Bin using one of the following methods.

Recycle Bin console

To view snapshots in the Recycle Bin using the console

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the snapshots that are currently in the Recycle Bin. To view the details for a specific snapshot, select it in the grid and choose **Actions**, **View details**.

AWS CLI

To view snapshots in the Recycle Bin using the AWS CLI

Use the list-snapshots-in-recycle-bin AWS CLI command. Include the `--snapshot-id` option to view a specific snapshot. Or omit the `--snapshot-id` option to view all snapshots in the Recycle Bin.

```
aws ec2 list-snapshots-in-recycle-bin --snapshot-id snapshot_id
```

For example, the following command provides information about snapshot `snap-01234567890abcdef` in the Recycle Bin.

```
aws ec2 list-snapshots-in-recycle-bin --snapshot-id snap-01234567890abcdef
```

Example output:

```
{
    "SnapshotRecycleBinInfo": [
        {
            "Description": "Monthly data backup snapshot",
            "RecycleBinEnterTime": "2021-12-01T13:00:00.000Z",
            "RecycleBinExitTime": "2021-12-15T13:00:00.000Z",
            "VolumeId": "vol-abcdef09876543210",
            "SnapshotId": "snap-01234567890abcdef"
        }
    ]
}
```

Restore snapshots from the Recycle Bin

You can't use a snapshot in any way while it is in the Recycle Bin. To use the snapshot, you must first restore it. When you restore a snapshot from the Recycle Bin, the snapshot is immediately available for use, and it is removed from the Recycle Bin. You can use a restored snapshot in the same way that you use any other snapshot in your account.

You can restore a snapshot from the Recycle Bin using one of the following methods.

Recycle Bin console

To restore a snapshot from the Recycle Bin using the console

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the snapshots that are currently in the Recycle Bin. Select the snapshot to restore and and choose **Recover**.
4. When prompted, choose **Recover**.

AWS CLI

To restore a deleted snapshot from the Recycle Bin using the AWS CLI

Use the restore-snapshot-from-recycle-bin AWS CLI command. For `--snapshot-id`, specify the ID of the snapshot to restore.

```
aws ec2 restore-snapshot-from-recycle-bin --snapshot-id snapshot_id
```

For example, the following command restores snapshot `snap-01234567890abcdef` from the Recycle Bin.

```
aws ec2 restore-snapshot-from-recycle-bin --snapshot-id snap-01234567890abcdef
```

Example output:

```
{
    "SnapshotId": "snap-01234567890abcdef",
    "Description": "Monthly data backup snapshot",
    "Encrypted": false,
    "OwnerId": "111122223333",
    "Progress": "100%",
    "StartTime": "2021-12-01T13:00:00.000000+00:00",
    "State": "recovering",
    "VolumeId": "vol-ffffffff",
    "VolumeSize": 30
}
```

Recover deleted volumes from the Recycle Bin

This topic explains how to recover Amazon EBS volumes from the Recycle Bin.

Topics

Permissions for working with volumes in the Recycle Bin

By default, users don't have permission to work with volumes that are in the Recycle Bin. To allow users to work with these resources, you must create IAM policies that grant permission to use specific resources and API actions. After the policies are created, you must add permissions to your users, groups, or roles.

To view and recover volumes that are in the Recycle Bin, users must have the following permissions:

- `ec2:ListVolumesInRecycleBin`
- `ec2:RestoreVolumeFromRecycleBin`

To manage tags for volumes in the Recycle Bin, users need the following additional permissions.

- `ec2:CreateTags`
- `ec2:DeleteTags`

To use the Recycle Bin console, users need the `ec2:DescribeTags` permission.

The following is an example IAM policy. It includes the `ec2:DescribeTags` permission for console users, and it includes the `ec2:CreateTags` and `ec2:DeleteTags` permissions for managing tags. If the permissions are not needed, you can remove them from the policy.

```
{
  "Version": "2012-10-17",
  "Statement": [
```

```
    {
      "Sid": "AllowRecycleBinVolumeOperations",
      "Effect": "Allow",
      "Action": [
        "ec2:ListVolumesInRecycleBin",
        "ec2:RestoreVolumeFromRecycleBin"
      ],
      "Resource": "arn:aws:ec2:*:123456789012:volume/*"
    },
    {
      "Sid": "AllowVolumeTagOperations",
      "Effect": "Allow",
      "Action": [
        "ec2:CreateTags",
        "ec2:DeleteTags",
        "ec2:DescribeTags"
      ],
      "Resource": "arn:aws:ec2:*:123456789012:volume/*"
    }
  ]
}
```

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

For more information about the permissions needed to use Recycle Bin, see Permissions for working with Recycle Bin and retention rules.

View volumes in the Recycle Bin

While a volume is in the Recycle Bin, you can view limited information about it, including:

- The ID of the volume.
- The size of the volume.
- The volume type.
- The date and time when the volume was deleted and it entered Recycle Bin.
- The date and time when the retention period expires. The volume will be permanently deleted from the Recycle Bin at this time.

You can view the volumes in the Recycle Bin using one of the following methods.

Recycle Bin console

To view volumes in the Recycle Bin using the console

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the volumes that are currently in the Recycle Bin. To view the details for a specific volume, select it in the grid and choose **Actions**, **View details**.

AWS CLI

To view volumes in the Recycle Bin using the AWS CLI

Use the list-volumes-in-recycle-bin AWS CLI command. Include the `--volume-id` option to view a specific volume. Or omit the `--volume-id` option to view all volumes in the Recycle Bin.

```
aws ec2 list-volumes-in-recycle-bin --volume-id volume_id
```

For example, the following command provides information about volume `vol-01234567890abcdef` in the Recycle Bin.

```
aws ec2 list-volumes-in-recycle-bin --volume-id vol-01234567890abcdef
```

Example output:

```
{
    "VolumeRecycleBinInfo": [
        {
            "VolumeId": "vol-01234567890abcdef",
            "RecycleBinEnterTime": "2021-12-01T13:00:00.000Z",
            "RecycleBinExitTime": "2021-12-08T13:00:00.000Z"
        }
    ]
}
```

Restore volumes from the Recycle Bin

You can't use a volume in any way while it is in the Recycle Bin. To use the volume, you must first restore it. When you restore a volume from the Recycle Bin, the volume is immediately available for use, and it is removed from the Recycle Bin. You can use a restored volume in the same way that you use any other volume in your account.

You can restore a volume from the Recycle Bin using one of the following methods.

Recycle Bin console

To restore a volume from the Recycle Bin using the console

1. Open the Recycle Bin console at https://console.aws.amazon.com/rbin/home/
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the volumes that are currently in the Recycle Bin. Select the volume to restore and choose **Recover**.
4. When prompted, choose **Recover**.

AWS CLI

To restore a deleted volume from the Recycle Bin using the AWS CLI

Use the restore-volume-from-recycle-bin AWS CLI command. For `--volume-id`, specify the ID of the volume to restore.

```
aws ec2 restore-volume-from-recycle-bin --volume-id volume_id
```

For example, the following command restores volume `vol-01234567890abcdef` from the Recycle Bin.

```
aws ec2 restore-volume-from-recycle-bin --volume-id vol-01234567890abcdef
```

Example output:

```
{
    "VolumeId": "vol-01234567890abcdef",
    "State": "available",
    "Size": 100,
    "VolumeType": "gp3",
    "AvailabilityZone": "us-east-1a",
    "CreateTime": "2021-12-01T13:00:00.000000+00:00",
    "Encrypted": false
}
```

Recover deleted AMIs from the Recycle Bin

This topic explains how to recover Amazon EBS-backed AMIs from the Recycle Bin.

Topics

Permissions for working with AMIs in the Recycle Bin

By default, users don't have permission to work with AMIs that are in the Recycle Bin. To allow users to work with these resources, you must create IAM policies that grant permission to use specific resources and API actions. After the policies are created, you must add permissions to your users, groups, or roles.

To view and recover AMIs that are in the Recycle Bin, users must have the following permissions:

- `ec2:ListImagesInRecycleBin`
- `ec2:RestoreImageFromRecycleBin`

To manage tags for AMIs in the Recycle Bin, users need the following additional permissions.

- `ec2:CreateTags`
- `ec2:DeleteTags`

To use the Recycle Bin console, users need the `ec2:DescribeTags` permission.

The following is an example IAM policy. It includes the `ec2:DescribeTags` permission for console users, and it includes the `ec2:CreateTags` and `ec2:DeleteTags` permissions for managing tags. If the permissions are not needed, you can remove them from the policy.

To provide access, add permissions to your users, groups, or roles:

- Users and groups in AWS IAM Identity Center:

  Create a permission set. Follow the instructions in Create a permission set in the *AWS IAM Identity Center User Guide*.
- Users managed in IAM through an identity provider:

  Create a role for identity federation. Follow the instructions in Create a role for a third-party identity provider (federation) in the *IAM User Guide*.
- IAM users:
  - Create a role that your user can assume. Follow the instructions in Create a role for an IAM user in the *IAM User Guide*.
  - (Not recommended) Attach a policy directly to a user or add a user to a user group. Follow the instructions in Adding permissions to a user (console) in the *IAM User Guide*.

For more information about the permissions needed to use Recycle Bin, see Permissions for working with Recycle Bin and retention rules.

View AMIs in the Recycle Bin

While an AMI is in the Recycle Bin, you can view limited information about it, including:

- The name, description, and unique ID of the AMI.
- The date and time when the AMI was deleted and it entered Recycle Bin.
- The date and time when the retention period expires. The AMI will be permanently deleted at this time.

You can view the AMIs in the Recycle Bin using one of the following methods.

Recycle Bin console

To view deleted AMIs in the Recycle Bin using the console

1. Open the Recycle Bin console at console.aws.amazon.com/rbin/home/.
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the resources that are currently in the Recycle Bin. To view the details for a specific AMI, select it in the grid, and choose **Actions**, **View details**.

AWS CLI

To view deleted AMIs in the Recycle Bin using the AWS CLI

Use the list-images-in-recycle-bin AWS CLI command. To view specific AMIs, include the `--image-id` option and specify the IDs of the AMIs to view. You can specify up to 20 IDs in a single request.

To view all of the AMIs in the Recycle Bin, omit the `--image-id` option. If you do not specify a value for `--max-items`, the command returns 1,000 items per page, by default. For more information, see Pagination in the *Amazon EC2 API Reference*.

```
aws ec2 list-images-in-recycle-bin --image-id ami_id
```

For example, the following command provides information about AMI `ami-01234567890abcdef` in the Recycle Bin.

```
aws ec2 list-images-in-recycle-bin --image-id ami-01234567890abcdef
```

Example output:

```
{
    "Images": [
        {
            "ImageId": "ami-0f740206c743d75df",
            "Name": "My AL2 AMI",
            "Description": "My Amazon Linux 2 AMI",
            "RecycleBinEnterTime": "2021-11-26T21:04:50+00:00",
```

```
            "RecycleBinExitTime": "2022-03-06T21:04:50+00:00"
        }
    ]
}
```

> **Important**
>
> If you receive the following error, you might need to update your AWS CLI version. For more information, see Command not found errors .
>
> ```
> aws.exe: error: argument operation: Invalid choice, valid choices are: ...
> ```

Restore AMIs from the Recycle Bin

You can't use an AMI in any way while it is in the Recycle Bin. To use the AMI, you must first restore it. When you restore an AMI from the Recycle Bin, the AMI is immediately available for use, and it is removed from the Recycle Bin. You can use a restored AMI in the same way that you use any other AMI in your account.

You can restore an AMI from the Recycle Bin using one of the following methods.

Recycle Bin console

To restore an AMI from the Recycle Bin using the console

1. Open the Recycle Bin console at console.aws.amazon.com/rbin/home/.
2. In the navigation pane, choose **Recycle Bin**.
3. The grid lists all of the resources that are currently in the Recycle Bin. Select the AMI to restore, and choose **Recover**.
4. When prompted, choose **Recover**.

AWS CLI

To restore a deleted AMI from the Recycle Bin using the AWS CLI

Use the restore-image-from-recycle-bin AWS CLI command. For `--image-id`, specify the ID of the AMI to restore.

```
aws ec2 restore-image-from-recycle-bin --image-id ami_id
```

For example, the following command restores AMI `ami-01234567890abcdef` from the Recycle Bin.

```
aws ec2 restore-image-from-recycle-bin --image-id ami-01234567890abcdef
```

The command returns no output on success.

> **Important**
>
> If you receive the following error, you might need to update your AWS CLI version. For more information, see Command not found errors .
>
> ```
> aws.exe: error: argument operation: Invalid choice, valid choices are: ...
> ```

Monitor Recycle Bin using Amazon EventBridge

Recycle Bin sends events to Amazon EventBridge for actions performed on retention rules. With EventBridge, you can establish rules that initiate programmatic actions in response to these events. For example, you can create a EventBridge rule that sends a notification to your email when a retention rule is unlocked and it enters its unlock delay period. For more information, see Creating Amazon EventBridge rules that react to events.

Events in EventBridge are represented as JSON objects. The fields that are unique to the event are contained in the `detail` section of the JSON object. The `event` field contains the event name. The `result` field contains the completed status of the action that initiated the event. For more information, see Amazon EventBridge event patterns in the *Amazon EventBridge User Guide*.

For more information about Amazon EventBridge, see What Is Amazon EventBridge? in the *Amazon EventBridge User Guide*.

Events

- RuleLocked
- RuleChangeAttempted

- [RuleUnlockScheduled]
- [RuleUnlockingNotice]
- [RuleUnlocked]

RuleLocked

The following is an example of an event that Recycle Bin generates when a retention rule is successfully locked. This event can be generated by **CreateRule** and **LockRule** requests. The API that generated the event is noted in the `api-name` field.

```
{
 "version": "0",
 "id": "exampleb-b491-4cf7-a9f1-bf370example",
 "detail-type": "Recycle Bin Rule Locked",
 "source": "aws.rbin",
 "account": "123456789012",
 "time": "2022-08-10T16:37:50Z",
 "region": "us-west-2",
 "resources": [
 "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
 ],
 "detail":
 {
 "detail-version": " 1.0.0",
 "rule-id": "a12345abcde",
 "rule-description": "locked account level rule",
 "unlock-delay-period": "30 days",
 "api-name": "CreateRule"
 }
 }
```

RuleChangeAttempted

The following is an example of an event that Recycle Bin generates for unsuccessful attempts to modify or delete a locked rule. This event can be generated by **DeleteRule** and **UpdateRule** requests. The API that generated the event is noted in the `api-name` field.

```
{
 "version": "0",
```

```
"id": "exampleb-b491-4cf7-a9f1-bf370example",
"detail-type": "Recycle Bin Rule Change Attempted",
"source": "aws.rbin",
"account": "123456789012",
"time": "2022-08-10T16:37:50Z",
"region": "us-west-2",
"resources": [
"arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
],
"detail":
{
"detail-version": " 1.0.0",
"rule-id": "a12345abcde",
"rule-description": "locked account level rule",
"unlock-delay-period": "30 days",
"api-name": "DeleteRule"
}
}
```

RuleUnlockScheduled

The following is an example of an event that Recycle Bin generates when a retention rule is unlocked and it starts its unlock delay period.

```
{
"version": "0",
"id": "exampleb-b491-4cf7-a9f1-bf370example",
"detail-type": "Recycle Bin Rule Unlock Scheduled",
"source": "aws.rbin",
"account": "123456789012",
"time": "2022-08-10T16:37:50Z",
"region": "us-west-2",
"resources": [
"arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
],
"detail":
{
"detail-version": " 1.0.0",
"rule-id": "a12345abcde",
"rule-description": "locked account level rule",
"unlock-delay-period": "30 days",
"scheduled-unlock-time": "2022-09-10T16:37:50Z",
}
```

```
}
```

RuleUnlockingNotice

The following is an example of an event that Recycle Bin generates daily while a retention rule is in its unlock delay period, until the day before the unlock delay period expires.


```
{
 "version": "0",
 "id": "exampleb-b491-4cf7-a9f1-bf370example",
 "detail-type": "Recycle Bin Rule Unlocking Notice",
 "source": "aws.rbin",
 "account": "123456789012",
 "time": "2022-08-10T16:37:50Z",
 "region": "us-west-2",
 "resources": [
 "arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
 ],
 "detail":
 {
 "detail-version": " 1.0.0",
 "rule-id": "a12345abcde",
 "rule-description": "locked account level rule",
 "unlock-delay-period": "30 days",
 "scheduled-unlock-time": "2022-09-10T16:37:50Z"
 }
 }
```


RuleUnlocked

The following is an example of an event that Recycle Bin generates when the unlock delay period for a retention rule expires and the retention rule can be modified or deleted.


```
{
 "version": "0",
 "id": "exampleb-b491-4cf7-a9f1-bf370example",
 "detail-type": "Recycle Bin Rule Unlocked",
 "source": "aws.rbin",
 "account": "123456789012",
 "time": "2022-08-10T16:37:50Z",
 "region": "us-west-2",
```

```
"resources": [
"arn:aws:rbin:us-west-2:123456789012:rule/a12345abcde"
],
"detail":
{
"detail-version": " 1.0.0",
"rule-id": "a12345abcde",
"rule-description": "locked account level rule",
"unlock-delay-period": "30 days",
"scheduled-unlock-time": "2022-09-10T16:37:50Z"
}
}
```

Monitor Recycle Bin using AWS CloudTrail

The Recycle Bin service is integrated with AWS CloudTrail. CloudTrail is a service that provides a record of actions taken by a user, role, or an AWS service. CloudTrail captures all API calls performed in Recycle Bin as events. If you create a trail, you can enable continuous delivery of CloudTrail events to an Amazon Simple Storage Service (Amazon S3) bucket. If you don't configure a trail, you can still view the most recent management events in the CloudTrail console in **Event history**. You can use the information collected by CloudTrail to determine the request that was made to Recycle Bin, the IP address from which the request was made, who made the request, when it was made, and additional details.

For more information about CloudTrail, see the AWS CloudTrail User Guide.

Recycle Bin information in CloudTrail

CloudTrail is enabled on your AWS account when you create the account. When supported event activity occurs in Recycle Bin, that activity is recorded in a CloudTrail event along with other AWS service events in **Event history**. You can view, search, and download recent events in your AWS account. For more information, see Viewing Events with CloudTrail Event History.

For an ongoing record of events in your AWS account, including events for Recycle Bin, create a trail. A *trail* enables CloudTrail to deliver log files to an S3 bucket. By default, when you create a trail in the console, the trail applies to all AWS Regions. The trail logs events from all Regions in the AWS partition and delivers the log files to the S3 bucket that you specify. Additionally, you can configure other AWS services to further analyze and act upon the event data collected in CloudTrail logs. For more information, see Overview for creating a trail in the *AWS CloudTrail User Guide*.

Supported API actions

For Recycle Bin, you can use CloudTrail to log the following API actions as *management events.*

- CreateRule
- UpdateRule
- GetRules
- ListRule
- DeleteRule
- TagResource
- UntagResource
- ListTagsForResource
- LockRule
- UnlockRule

For more information about logging management events, see Logging management events for trails in the *CloudTrail User Guide*.

Identity information

Every event or log entry contains information about who generated the request. The identity information helps you determine the following:

- Whether the request was made with root user or user credentials.
- Whether the request was made with temporary security credentials for a role or federated user.
- Whether the request was made by another AWS service.

For more information, see the CloudTrail userIdentityElement.

Understand Recycle Bin log file entries

A trail is a configuration that enables delivery of events as log files to an S3 bucket that you specify. CloudTrail log files contain one or more log entries. An event represents a single request from any source and includes information about the requested action, the date and time of the action, request parameters, and so on. CloudTrail log files aren't an ordered stack trace of the public API calls, so they don't appear in any specific order.

The following are example CloudTrail log entries.

CreateRule


```
{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "123456789012",
 "arn": "arn:aws:iam::123456789012:root",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {
   "sessionIssuer": {
     "type": "Role",
     "principalId": "123456789012",
     "arn": "arn:aws:iam::123456789012:role/Admin",
     "accountId": "123456789012",
     "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
     "mfaAuthenticated": "false",
     "creationDate": "2021-08-02T21:43:38Z"
   }
 }
 },
 "eventTime": "2021-08-02T21:45:22Z",
 "eventSource": "rbin.amazonaws.com",
 "eventName": "CreateRule",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "123.123.123.123",
 "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
 "requestParameters": {
 "retentionPeriod": {
   "retentionPeriodValue": 7,
   "retentionPeriodUnit": "DAYS"
 },
 "description": "Match all snapshots",
 "resourceType": "EBS_SNAPSHOT"
 },
 "responseElements": {
 "identifier": "jkrnexample"
```

```
},
"requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
"eventID": "714fafex-2eam-42pl-913e-926d4example",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```

GetRule

```
{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "123456789012",
 "arn": "arn:aws:iam::123456789012:root",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {
   "sessionIssuer": {
     "type": "Role",
     "principalId": "123456789012",
     "arn": "arn:aws:iam::123456789012:role/Admin",
     "accountId": "123456789012",
     "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
     "mfaAuthenticated": "false",
     "creationDate": "2021-08-02T21:43:38Z"
   }
 }
 },
 "eventTime": "2021-08-02T21:45:33Z",
 "eventSource": "rbin.amazonaws.com",
 "eventName": "GetRule",
```

```
"awsRegion": "us-west-2",
"sourceIPAddress": "123.123.123.123",
"userAgent": "aws-cli/1.20.9 Python/3.6.14
Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
"requestParameters": {
"identifier": "jkrnexample"
},
"responseElements": null,
"requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
"eventID": "714fafex-2eam-42pl-913e-926d4example",
"readOnly": true,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```


ListRules


```
{
"eventVersion": "1.08",
"userIdentity": {
"type": "AssumedRole",
"principalId": "123456789012",
"arn": "arn:aws:iam::123456789012:root",
"accountId": "123456789012",
"accessKeyId": "AKIAIOSFODNN7EXAMPLE",
"sessionContext": {
  "sessionIssuer": {
    "type": "Role",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:role/Admin",
    "accountId": "123456789012",
    "userName": "Admin"
  },
  "webIdFederationData": {},
  "attributes": {
    "mfaAuthenticated": "false",
```

```
    "creationDate": "2021-08-02T21:43:38Z"
  }
 }
 },
 "eventTime": "2021-08-02T21:44:37Z",
 "eventSource": "rbin.amazonaws.com",
 "eventName": "ListRules",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "123.123.123.123",
 "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
 "requestParameters": {
 "resourceTags": [
   {
     "resourceTagKey": "test",
     "resourceTagValue": "test"
   }
 ]
 },
 "responseElements": null,
 "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
 "eventID": "714fafex-2eam-42pl-913e-926d4example",
 "readOnly": true,
 "eventType": "AwsApiCall",
 "managementEvent": true,
 "eventCategory": "Management",
 "recipientAccountId": "123456789012",
 "tlsDetails": {
 "tlsVersion": "TLSv1.2",
 "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
 "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
 }
 }
```


UpdateRule


```
{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "123456789012",
 "arn": "arn:aws:iam::123456789012:root",
 "accountId": "123456789012",
```

```
"accessKeyId": "AKIAIOSFODNN7EXAMPLE",
"sessionContext": {
  "sessionIssuer": {
    "type": "Role",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:role/Admin",
    "accountId": "123456789012",
    "userName": "Admin"
  },
  "webIdFederationData": {},
  "attributes": {
    "mfaAuthenticated": "false",
    "creationDate": "2021-08-02T21:43:38Z"
  }
}
},
"eventTime": "2021-08-02T21:46:03Z",
"eventSource": "rbin.amazonaws.com",
"eventName": "UpdateRule",
"awsRegion": "us-west-2",
"sourceIPAddress": "123.123.123.123",
"userAgent": "aws-cli/1.20.9 Python/3.6.14
Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
"requestParameters": {
"identifier": "jkrnexample",
"retentionPeriod": {
  "retentionPeriodValue": 365,
  "retentionPeriodUnit": "DAYS"
},
"description": "Match all snapshots",
"resourceType": "EBS_SNAPSHOT"
},
"responseElements": null,
"requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
"eventID": "714fafex-2eam-42pl-913e-926d4example",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
```

```
}
}
```

DeleteRule


```
{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "123456789012",
 "arn": "arn:aws:iam::123456789012:root",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {
   "sessionIssuer": {
     "type": "Role",
     "principalId": "123456789012",
     "arn": "arn:aws:iam::123456789012:role/Admin",
     "accountId": "123456789012",
     "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
     "mfaAuthenticated": "false",
     "creationDate": "2021-08-02T21:43:38Z"
   }
 }
 },
 "eventTime": "2021-08-02T21:46:25Z",
 "eventSource": "rbin.amazonaws.com",
 "eventName": "DeleteRule",
 "awsRegion": "us-west-2",
 "sourceIPAddress": "123.123.123.123",
 "userAgent": "aws-cli/1.20.9 Python/3.6.14
 Linux/4.9.230-0.1.ac.224.84.332.metal1.x86_64 botocore/1.21.9",
 "requestParameters": {
 "identifier": "jkrnexample"
 },
 "responseElements": null,
 "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
 "eventID": "714fafex-2eam-42pl-913e-926d4example",
 "readOnly": false,
 "eventType": "AwsApiCall",
```

```
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```

TagResource

```
{
"eventVersion": "1.08",
"userIdentity": {
"type": "AssumedRole",
"principalId": "123456789012",
"arn": "arn:aws:iam::123456789012:root",
"accountId": "123456789012",
"accessKeyId": "AKIAIOSFODNN7EXAMPLE",
"sessionContext": {
  "sessionIssuer": {
    "type": "Role",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:role/Admin",
    "accountId": "123456789012",
    "userName": "Admin"
  },
  "webIdFederationData": {},
  "attributes": {
    "mfaAuthenticated": "false",
    "creationDate": "2021-10-22T21:38:34Z"
  }
}
},
"eventTime": "2021-10-22T21:43:15Z",
"eventSource": "rbin.amazonaws.com",
"eventName": "TagResource",
"awsRegion": "us-west-2",
"sourceIPAddress": "123.123.123.123",
"userAgent": "aws-cli/1.20.26 Python/3.6.14
Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
"requestParameters": {
```

```
"resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234",
"tags": [
  {
    "key": "purpose",
    "value": "production"
  }
]
},
"responseElements": null,
"requestID": "examplee-7962-49ec-8633-795efexample",
"eventID": "example4-6826-4c0a-bdec-0bab1example",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```


UntagResource


```
{
 "eventVersion": "1.08",
 "userIdentity": {
 "type": "AssumedRole",
 "principalId": "123456789012",
 "arn": "arn:aws:iam::123456789012:root",
 "accountId": "123456789012",
 "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "sessionContext": {
   "sessionIssuer": {
     "type": "Role",
     "principalId": "123456789012",
     "arn": "arn:aws:iam::123456789012:role/Admin",
     "accountId": "123456789012",
     "userName": "Admin"
   },
   "webIdFederationData": {},
   "attributes": {
```

```
    "mfaAuthenticated": "false",
    "creationDate": "2021-10-22T21:38:34Z"
  }
}
},
"eventTime": "2021-10-22T21:44:16Z",
"eventSource": "rbin.amazonaws.com",
"eventName": "UntagResource",
"awsRegion": "us-west-2",
"sourceIPAddress": "123.123.123.123",
"userAgent": "aws-cli/1.20.26 Python/3.6.14
Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
"requestParameters": {
"resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234",
"tagKeys": [
  "purpose"
]
},
"responseElements": null,
"requestID": "example7-6c1e-4f09-9e46-bb957example",
"eventID": "example6-75ff-4c94-a1cd-4d5f5example",
"readOnly": false,
"eventType": "AwsApiCall",
"managementEvent": true,
"eventCategory": "Management",
"recipientAccountId": "123456789012",
"tlsDetails": {
"tlsVersion": "TLSv1.2",
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```


ListTagsForResource


```
{
"eventVersion": "1.08",
"userIdentity": {
"type": "AssumedRole",
"principalId": "123456789012",
"arn": "arn:aws:iam::123456789012:root",
"accountId": "123456789012",
"accessKeyId": "AKIAIOSFODNN7EXAMPLE",
```

```
  "sessionContext": {
    "sessionIssuer": {
      "type": "Role",
      "principalId": "123456789012",
      "arn": "arn:aws:iam::123456789012:role/Admin",
      "accountId": "123456789012",
      "userName": "Admin"
    },
    "webIdFederationData": {},
    "attributes": {
      "mfaAuthenticated": "false",
      "creationDate": "2021-10-22T21:38:34Z"
    }
  }
  },
  "eventTime": "2021-10-22T21:42:31Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "ListTagsForResource",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "aws-cli/1.20.26 Python/3.6.14
  Linux/4.9.273-0.1.ac.226.84.332.metal1.x86_64 botocore/1.21.26",
  "requestParameters": {
  "resourceArn": "arn:aws:rbin:us-west-2:123456789012:rule/ABCDEF01234"
  },
  "responseElements": null,
  "requestID": "example8-10c7-43d4-b147-3d9d9example",
  "eventID": "example2-24fc-4da7-a479-c9748example",
  "readOnly": true,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "eventCategory": "Management",
  "recipientAccountId": "123456789012",
  "tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
  }
```

LockRule

```
{
```

```
"eventVersion": "1.08",
"userIdentity": {
"type": "AssumedRole",
"principalId": "123456789012",
"arn": "arn:aws:iam::123456789012:root",
"accountId": "123456789012",
"accessKeyId": "AKIAIOSFODNN7EXAMPLE",
"sessionContext": {
  "sessionIssuer": {
    "type": "Role",
    "principalId": "123456789012",
    "arn": "arn:aws:iam::123456789012:role/Admin",
    "accountId": "123456789012",
    "userName": "Admin"
  },
  "webIdFederationData": {},
  "attributes": {
    "creationDate": "2022-10-25T00:45:11Z",
    "mfaAuthenticated": "false"
  }
}
},
"eventTime": "2022-10-25T00:45:19Z",
"eventSource": "rbin.amazonaws.com",
"eventName": "LockRule",
"awsRegion": "us-west-2",
"sourceIPAddress": "123.123.123.123",
"userAgent": "python-requests/2.25.1",
"requestParameters": {
"identifier": "jkrnexample",
"lockConfiguration": {
  "unlockDelay": {
    "unlockDelayValue": 7,
    "unlockDelayUnit": "DAYS"
  }
}
},
"responseElements": {
"identifier": "jkrnexample",
"description": "",
"resourceType": "EBS_SNAPSHOT",
"retentionPeriod": {
  "retentionPeriodValue": 7,
  "retentionPeriodUnit": "DAYS"
```

```
  },
  "resourceTags": [],
  "status": "available",
  "lockConfiguration": {
    "unlockDelay": {
      "unlockDelayValue": 7,
      "unlockDelayUnit": "DAYS"
    }
  },
  "lockState": "locked"
  },
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management",
  "tlsDetails": {
  "tlsVersion": "TLSv1.2",
  "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
  "clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
  }
  }
```


UnlockRule


```
{
  "eventVersion": "1.08",
  "userIdentity": {
  "type": "AssumedRole",
  "principalId": "123456789012",
  "arn": "arn:aws:iam::123456789012:root",
  "accountId": "123456789012",
  "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
  "sessionContext": {
    "sessionIssuer": {
      "type": "Role",
      "principalId": "123456789012",
      "arn": "arn:aws:iam::123456789012:role/Admin",
      "accountId": "123456789012",
      "userName": "Admin"
    },
```

```
    "webIdFederationData": {},
    "attributes": {
      "creationDate": "2022-10-25T00:45:11Z",
      "mfaAuthenticated": "false"
    }
  }
  },
  "eventTime": "2022-10-25T00:46:17Z",
  "eventSource": "rbin.amazonaws.com",
  "eventName": "UnlockRule",
  "awsRegion": "us-west-2",
  "sourceIPAddress": "123.123.123.123",
  "userAgent": "python-requests/2.25.1",
  "requestParameters": {
  "identifier": "jkrnexample"
  },
  "responseElements": {
  "identifier": "jkrnexample",
  "description": "",
  "resourceType": "EC2_IMAGE",
  "retentionPeriod": {
    "retentionPeriodValue": 7,
    "retentionPeriodUnit": "DAYS"
  },
  "resourceTags": [],
  "status": "available",
  "lockConfiguration": {
    "unlockDelay": {
      "unlockDelayValue": 7,
      "unlockDelayUnit": "DAYS"
    }
  },
  "lockState": "pending_unlock",
  "lockEndTime": "Nov 1, 2022, 12:46:17 AM"
  },
  "requestID": "ex0577a5-amc4-pl4f-ef51-50fdexample",
  "eventID": "714fafex-2eam-42pl-913e-926d4example",
  "readOnly": false,
  "eventType": "AwsApiCall",
  "managementEvent": true,
  "recipientAccountId": "123456789012",
  "eventCategory": "Management",
  "tlsDetails": {
  "tlsVersion": "TLSv1.2",
```

```
"cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
"clientProvidedHostHeader": "rbin.us-west-2.amazonaws.com"
}
}
```

Service endpoints for Recycle Bin

An *endpoint* is a URL that serves as an entry point for an AWS web service. Recycle Bin supports the following endpoint types:

- IPv4 endpoints
- Dual-stack endpoints that support both IPv4 and IPv6
- FIPS endpoints

When you make a request, you can specify the endpoint and Region to use. If you do not specify an endpoint, the IPv4 endpoint is used by default. To use a different endpoint type, you must specify it in your request. For examples of how to do this, see Specifying endpoints.

For the Recycle Bin, see Recycle Bin endpoints in the *Amazon Web Services General Reference*.

Topics

- IPv4 endpoints
- Dual-stack (IPv4 and IPv6) endpoints
- FIPS endpoints
- Specifying endpoints

IPv4 endpoints

IPv4 endpoints support IPv4 traffic only. IPv4 endpoints are available for all Regions.

You must specify the Region as part of the endpoint name. The endpoint names use the following naming convention:

- rbin.*region*.amazonaws.com

For example, the IPv4 endpoint for the US East (N. Virginia) Region is `rbin.us-east-1.amazonaws.com`.

Dual-stack (IPv4 and IPv6) endpoints

Dual-stack endpoints support both IPv4 and IPv6 traffic. Dual-stack endpoints are available for all Regions.

To use IPv6, you must use a dual-stack endpoint. When you make a request to a dual-stack endpoint, the endpoint URL resolves to an IPv6 or an IPv4 address, depending on the protocol used by your network and client.

You must specify the Region as part of the endpoint name. Dual-stack endpoint names use the following naming convention:

- `rbin.region.api.aws`

For example, the dual-stack endpoint for the US East (N. Virginia) Region is `rbin.us-east-1.api.aws`.

FIPS endpoints

Recycle Bin provides FIPS-validated IPv4 and dual-stack (IPv4 and IPv6) endpoints for the following Regions:

- `us-east-1` — US East (N. Virginia)
- `us-east-2` — US East (Ohio)
- `us-west-1` — US West (N. California)
- `us-west-2` — US West (Oregon)
- `ca-central-1` — Canada (Central)
- `ca-west-1` — Canada West (Calgary)
- `us-gov-east-1` — AWS GovCloud (US-East)
- `us-gov-west-1` — AWS GovCloud (US-West)

FIPS IPv4 endpoints use the following naming convention: `rbin-fips.region.amazonaws.com`. For example, the FIPS IPv4 endpoint for the US East (N. Virginia) Region is `rbin-fips.us-east-1.amazonaws.com`.

FIPS dual-stack endpoints use the following naming convention: `rbin-fips.region.api.aws`. For example, the FIPS dual-stack endpoint for the US East (N. Virginia) Region is `rbin-fips.us-east-1.api.aws`.

Specifying endpoints

The following examples show how to specify an endpoint for the `us-east-2` Region using the AWS CLI.

- **Dual-stack**

```
aws rbin get-rule \
--identifier rule_id \
--endpoint-url https://rbin.us-east-2.api.aws
```

- **IPv4**

```
aws rbin get-rule \
--identifier rule_id \
--endpoint-url https://rbin.us-east-2.amazonaws.com
```

Create a private connection between a VPC and Recycle Bin

You can establish a private connection between your VPC and Recycle Bin by creating an interface VPC endpoint, powered by AWS PrivateLink. You can access Recycle Bin as if it were in your VPC, without using an internet gateway, NAT device, VPN connection, or Direct Connect connection. Instances in your VPC don't need public IP addresses to communicate with Recycle Bin.

We create an endpoint network interface in each subnet that you enable for the interface endpoint.

For more information, see Access AWS services through AWS PrivateLink in the *AWS PrivateLink Guide*.

Create an interface VPC endpoint for Recycle Bin

You can create a VPC endpoint for Recycle Bin using either the Amazon VPC console or the AWS CLI. For more information, see Create a VPC endpoint in the *AWS PrivateLink Guide*.

Create a VPC endpoint for Recycle Bin using the following service name: `com.amazonaws.region.rbin`

If you enable private DNS for the endpoint, you can make API requests to Recycle Bin using its default DNS name for the Region, for example, `rbin.us-east-1.amazonaws.com`.

Create a VPC endpoint policy for Recycle Bin

By default, full access to Recycle Bin is allowed through the endpoint. You can control access to the interface endpoint using VPC endpoint policies. You can attach an endpoint policy to your VPC endpoint that controls access to Recycle Bin. The policy specifies the following information:

- The **principal** that can perform actions.
- The **actions** that can be performed.
- The **resources** on which actions can be performed.

For more information, see Controlling access to services with VPC endpoints in the *Amazon VPC User Guide*.

```
{
    "Statement": [
    {
        "Effect": "Allow",
        "Action": "rbin:*",
        "Resource": "*",
        "Principal": "*"
    },
    {
        "Effect": "Deny",
        "Action": "rbin:DeleteRule",
        "Resource": "*",
        "Principal": "*",
        "Condition": {
            "StringEquals" : {
                "rbin:Attribute/ResourceType": "EBS_SNAPSHOT"
            }
        }
    }]
}
```

Security in Amazon EBS

Cloud security at AWS is the highest priority. As an AWS customer, you benefit from data centers and network architectures that are built to meet the requirements of the most security-sensitive organizations.

Security is a shared responsibility between AWS and you. The shared responsibility model describes this as security *of* the cloud and security *in* the cloud:

- **Security of the cloud** – AWS is responsible for protecting the infrastructure that runs AWS services in the AWS Cloud. AWS also provides you with services that you can use securely. Third-party auditors regularly test and verify the effectiveness of our security as part of the AWS Compliance Programs. To learn about the compliance programs that apply to Amazon Elastic Block Store, see AWS Services in Scope by Compliance Program.
- **Security in the cloud** – Your responsibility is determined by the AWS service that you use. You are also responsible for other factors including the sensitivity of your data, your company's requirements, and applicable laws and regulations.

This documentation helps you understand how to apply the shared responsibility model when using Amazon EBS. The following topics show you how to configure Amazon EBS to meet your security and compliance objectives. You also learn how to use other AWS services that help you to monitor and secure your Amazon EBS resources.

Topics

Data protection in Amazon EBS

The AWS shared responsibility model applies to data protection in Amazon Elastic Block Store. As described in this model, AWS is responsible for protecting the global infrastructure that runs all of the AWS Cloud. You are responsible for maintaining control over your content that is hosted on this infrastructure. You are also responsible for the security configuration and management tasks for the AWS services that you use. For more information about data privacy, see the Data Privacy FAQ. For information about data protection in Europe, see the AWS Shared Responsibility Model and GDPR blog post on the *AWS Security Blog*.

For data protection purposes, we recommend that you protect AWS account credentials and set up individual users with AWS IAM Identity Center or AWS Identity and Access Management (IAM). That way, each user is given only the permissions necessary to fulfill their job duties. We also recommend that you secure your data in the following ways:

- Use multi-factor authentication (MFA) with each account.
- Use SSL/TLS to communicate with AWS resources. We require TLS 1.2 and recommend TLS 1.3.
- Set up API and user activity logging with AWS CloudTrail. For information about using CloudTrail trails to capture AWS activities, see Working with CloudTrail trails in the *AWS CloudTrail User Guide*.
- Use AWS encryption solutions, along with all default security controls within AWS services.
- Use advanced managed security services such as Amazon Macie, which assists in discovering and securing sensitive data that is stored in Amazon S3.
- If you require FIPS 140-3 validated cryptographic modules when accessing AWS through a command line interface or an API, use a FIPS endpoint. For more information about the available FIPS endpoints, see Federal Information Processing Standard (FIPS) 140-3.

We strongly recommend that you never put confidential or sensitive information, such as your customers' email addresses, into tags or free-form text fields such as a **Name** field. This includes when you work with Amazon EBS or other AWS services using the console, API, AWS CLI, or AWS SDKs. Any data that you enter into tags or free-form text fields used for names may be used for billing or diagnostic logs. If you provide a URL to an external server, we strongly recommend that you do not include credentials information in the URL to validate your request to that server.

Topics

Amazon EBS data security

Amazon EBS volumes are presented to you as raw, unformatted block devices. These devices are logical devices that are created on the EBS infrastructure and the Amazon EBS service ensures that the devices are logically empty (that is, the raw blocks are zeroed or they contain cryptographically pseudorandom data) prior to any use or re-use by a customer.

If you have procedures that require that all data be erased using a specific method, either after or before use (or both), such as those detailed in **DoD 5220.22-M** (National Industrial Security Program Operating Manual) or **NIST 800-88** (Guidelines for Media Sanitization), you have the ability to do so on Amazon EBS. That block-level activity will be reflected down to the underlying storage media within the Amazon EBS service.

Encryption at rest and in transit

Amazon EBS encryption is an encryption solution that enables you to encrypt your Amazon EBS volumes and Amazon EBS snapshots using AWS Key Management Service cryptographic keys. EBS encryption operations occur on the servers that host Amazon EC2 instances, ensuring the security of both **data-at-rest** and **data-in-transit** between an instance and its attached volume and any subsequent snapshots. For more information, see Amazon EBS encryption.

KMS key management

When you create an encrypted Amazon EBS volume or snapshot, you specify an AWS Key Management Service key. By default, Amazon EBS uses the AWS managed KMS key for Amazon EBS in your account and Region (`aws/ebs`). However, you can specify a customer managed KMS key that you create and manage. Using a customer managed KMS key gives you more flexibility, including the ability to create, rotate, and disable KMS keys.

To use a customer managed KMS key, you must give users permission to use the KMS key. For more information, see Permissions for users.

> **Important**
>
> Amazon EBS supports only symmetric KMS keys. You can't use asymmetric KMS keys to encrypt an Amazon EBS volume and snapshots. For help determining whether a KMS key is symmetric or asymmetric, see Identify asymmetric KMS keys.

For each volume, Amazon EBS asks AWS KMS to generate a unique data key encrypted under the KMS key that you specify. Amazon EBS stores the encrypted data key with the volume. Then, when you attach the volume to an Amazon EC2 instance, Amazon EBS calls AWS KMS to decrypt the data key. Amazon EBS uses the plaintext data key in hypervisor memory to encrypt all I/O to the volume. For more information, see How Amazon EBS encryption works.

Identity and access management for Amazon EBS

AWS Identity and Access Management (IAM) is an AWS service that helps an administrator securely control access to AWS resources. IAM administrators control who can be *authenticated* (signed in) and *authorized* (have permissions) to use Amazon EBS resources. IAM is an AWS service that you can use with no additional charge.

Topics

Audience

How you use AWS Identity and Access Management (IAM) differs based on your role:

- **Service user** - request permissions from your administrator if you cannot access features (see Troubleshoot Amazon EBS authorization issues)
- **Service administrator** - determine user access and submit permission requests (see How Amazon EBS works with IAM)
- **IAM administrator** - write policies to manage access (see Example IAM policies for Amazon EBS)

Authenticating with identities

Authentication is how you sign in to AWS using your identity credentials. You must be authenticated as the AWS account root user, an IAM user, or by assuming an IAM role.

You can sign in as a federated identity using credentials from an identity source like AWS IAM Identity Center (IAM Identity Center), single sign-on authentication, or Google/Facebook credentials. For more information about signing in, see How to sign in to your AWS account in the *AWS Sign-In User Guide*.

For programmatic access, AWS provides an SDK and CLI to cryptographically sign requests. For more information, see AWS Signature Version 4 for API requests in the *IAM User Guide*.

AWS account root user

When you create an AWS account, you begin with one sign-in identity called the AWS account *root user* that has complete access to all AWS services and resources. We strongly recommend that you don't use the root user for everyday tasks. For tasks that require root user credentials, see Tasks that require root user credentials in the *IAM User Guide*.

Federated identity

As a best practice, require human users to use federation with an identity provider to access AWS services using temporary credentials.

A *federated identity* is a user from your enterprise directory, web identity provider, or Directory Service that accesses AWS services using credentials from an identity source. Federated identities assume roles that provide temporary credentials.

For centralized access management, we recommend AWS IAM Identity Center. For more information, see What is IAM Identity Center? in the *AWS IAM Identity Center User Guide*.

IAM users and groups

An *IAM user* is an identity with specific permissions for a single person or application. We recommend using temporary credentials instead of IAM users with long-term credentials. For more information, see Require human users to use federation with an identity provider to access AWS using temporary credentials in the *IAM User Guide*.

An *IAM group* specifies a collection of IAM users and makes permissions easier to manage for large sets of users. For more information, see Use cases for IAM users in the *IAM User Guide*.

IAM roles

An *IAM role* is an identity with specific permissions that provides temporary credentials. You can assume a role by switching from a user to an IAM role (console) or by calling an AWS CLI or AWS API operation. For more information, see Methods to assume a role in the *IAM User Guide*.

IAM roles are useful for federated user access, temporary IAM user permissions, cross-account access, cross-service access, and applications running on Amazon EC2. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Managing access using policies

You control access in AWS by creating policies and attaching them to AWS identities or resources. A policy defines permissions when associated with an identity or resource. AWS evaluates these policies when a principal makes a request. Most policies are stored in AWS as JSON documents. For more information about JSON policy documents, see Overview of JSON policies in the *IAM User Guide*.

Using policies, administrators specify who has access to what by defining which **principal** can perform **actions** on what **resources**, and under what **conditions**.

By default, users and roles have no permissions. An IAM administrator creates IAM policies and adds them to roles, which users can then assume. IAM policies define permissions regardless of the method used to perform the operation.

Identity-based policies

Identity-based policies are JSON permissions policy documents that you attach to an identity (user, group, or role). These policies control what actions identities can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

Identity-based policies can be *inline policies* (embedded directly into a single identity) or *managed policies* (standalone policies attached to multiple identities). To learn how to choose between managed and inline policies, see Choose between managed policies and inline policies in the *IAM User Guide*.

Resource-based policies

Resource-based policies are JSON policy documents that you attach to a resource. Examples include IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. You must specify a principal in a resource-based policy.

Resource-based policies are inline policies that are located in that service. You can't use AWS managed policies from IAM in a resource-based policy.

Other policy types

AWS supports additional policy types that can set the maximum permissions granted by more common policy types:

- **Permissions boundaries** – Set the maximum permissions that an identity-based policy can grant to an IAM entity. For more information, see Permissions boundaries for IAM entities in the *IAM User Guide*.
- **Service control policies (SCPs)** – Specify the maximum permissions for an organization or organizational unit in AWS Organizations. For more information, see Service control policies in the *AWS Organizations User Guide*.
- **Resource control policies (RCPs)** – Set the maximum available permissions for resources in your accounts. For more information, see Resource control policies (RCPs) in the *AWS Organizations User Guide*.
- **Session policies** – Advanced policies passed as a parameter when creating a temporary session for a role or federated user. For more information, see Session policies in the *IAM User Guide*.

Multiple policy types

When multiple types of policies apply to a request, the resulting permissions are more complicated to understand. To learn how AWS determines whether to allow a request when multiple policy types are involved, see Policy evaluation logic in the *IAM User Guide*.

How Amazon EBS works with IAM

Before you use IAM to manage access to Amazon EBS, learn what IAM features are available to use with Amazon EBS.

IAM features you can use with Amazon Elastic Block Store

IAM feature	Amazon EBS support
Identity-based policies	Yes

IAM feature	Amazon EBS support
Resource-based policies	No
Policy actions	Yes
Policy resources	Yes
Policy condition keys	Yes
ACLs	No
ABAC (tags in policies)	Partial
Temporary credentials	Yes
Principal permissions	Yes
Service roles	Yes
Service-linked roles	No

To get a high-level view of how Amazon EBS and other AWS services work with most IAM features, see AWS services that work with IAM in the *IAM User Guide*.

Identity-based policies for Amazon EBS

Supports identity-based policies: Yes

Identity-based policies are JSON permissions policy documents that you can attach to an identity, such as an IAM user, group of users, or role. These policies control what actions users and roles can perform, on which resources, and under what conditions. To learn how to create an identity-based policy, see Define custom IAM permissions with customer managed policies in the *IAM User Guide*.

With IAM identity-based policies, you can specify allowed or denied actions and resources as well as the conditions under which actions are allowed or denied. To learn about all of the elements that you can use in a JSON policy, see IAM JSON policy elements reference in the *IAM User Guide*.

Identity-based policy examples for Amazon EBS

To view examples of Amazon EBS identity-based policies, see Example IAM policies for Amazon EBS.

Resource-based policies within Amazon EBS

Supports resource-based policies: No

Resource-based policies are JSON policy documents that you attach to a resource. Examples of resource-based policies are IAM *role trust policies* and Amazon S3 *bucket policies*. In services that support resource-based policies, service administrators can use them to control access to a specific resource. For the resource where the policy is attached, the policy defines what actions a specified principal can perform on that resource and under what conditions. You must specify a principal in a resource-based policy. Principals can include accounts, users, roles, federated users, or AWS services.

To enable cross-account access, you can specify an entire account or IAM entities in another account as the principal in a resource-based policy. For more information, see Cross account resource access in IAM in the *IAM User Guide*.

Policy actions for Amazon EBS

Supports policy actions: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Action` element of a JSON policy describes the actions that you can use to allow or deny access in a policy. Include actions in a policy to grant permissions to perform the associated operation.

To see a list of Amazon EBS actions, see Actions, resources, and condition keys for Amazon EC2 and Actions, resources, and condition keys for Amazon EBS in the *Service Authorization Reference*.

Policy actions in Amazon EBS use either the `ec2` or the `ebs` prefix before the action.

To specify multiple actions in a single statement, separate them with commas.

```
"Action": [
```

```
    "ec2:action1",
    "ec2:action2"
]
```

To view examples of Amazon EBS identity-based policies, see Example IAM policies for Amazon EBS.

Policy resources for Amazon EBS

Supports policy resources: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Resource` JSON policy element specifies the object or objects to which the action applies. As a best practice, specify a resource using its Amazon Resource Name (ARN). For actions that don't support resource-level permissions, use a wildcard (*) to indicate that the statement applies to all resources.

```
"Resource": "*"
```

Some Amazon EBS API actions support multiple resources. To specify multiple resources in a single statement, separate the ARNs with commas. For example, `DescribeVolumes` accesses vol-01234567890abcdef and vol-09876543210fedcba, so a principal must have permissions to access both resources.

```
"Resource": [
    "arn:aws:ec2:us-east-1:123456789012:volume/vol-01234567890abcdef",
    "arn:aws:ec2:us-east-1:123456789012:volume/vol-09876543210fedcba"
]
```

Policy condition keys for Amazon EBS

Supports service-specific policy condition keys: Yes

Administrators can use AWS JSON policies to specify who has access to what. That is, which **principal** can perform **actions** on what **resources**, and under what **conditions**.

The `Condition` element specifies when statements execute based on defined criteria. You can create conditional expressions that use condition operators, such as equals or less than, to match the condition in the policy with values in the request. To see all AWS global condition keys, see AWS global condition context keys in the *IAM User Guide*.

For example, the following condition allows the principal to perform an action on a volume only if the volume type is gp2.

```
"Condition":{
  "StringLikeIfExists":{
    "ec2:VolumeType":"gp2"
  }
}
```

To see a list of Amazon EBS condition keys, see Actions, resources, and condition keys in the *Service Authorization Reference*.

ACLs in Amazon EBS

Supports ACLs: No

Access control lists (ACLs) control which principals (account members, users, or roles) have permissions to access a resource. ACLs are similar to resource-based policies, although they do not use the JSON policy document format.

ABAC with Amazon EBS

Supports ABAC (tags in policies): Partial

Attribute-based access control (ABAC) is an authorization strategy that defines permissions based on attributes called tags. You can attach tags to IAM entities and AWS resources, then design ABAC policies to allow operations when the principal's tag matches the tag on the resource.

To control access based on tags, you provide tag information in the condition element of a policy using the `aws:ResourceTag/key-name`, `aws:RequestTag/key-name`, or `aws:TagKeys` condition keys.

If a service supports all three condition keys for every resource type, then the value is **Yes** for the service. If a service supports all three condition keys for only some resource types, then the value is **Partial**.

For more information about ABAC, see Define permissions with ABAC authorization in the *IAM User Guide*. To view a tutorial with steps for setting up ABAC, see Use attribute-based access control (ABAC) in the *IAM User Guide*.

Using temporary credentials with Amazon EBS

Supports temporary credentials: Yes

Temporary credentials provide short-term access to AWS resources and are automatically created when you use federation or switch roles. AWS recommends that you dynamically generate temporary credentials instead of using long-term access keys. For more information, see Temporary security credentials in IAM and AWS services that work with IAM in the *IAM User Guide*.

Cross-service principal permissions for Amazon EBS

Supports forward access sessions (FAS): Yes

Forward access sessions (FAS) use the permissions of the principal calling an AWS service, combined with the requesting AWS service to make requests to downstream services. For policy details when making FAS requests, see Forward access sessions.

Service roles for Amazon EBS

Supports service roles: Yes

A service role is an IAM role that a service assumes to perform actions on your behalf. An IAM administrator can create, modify, and delete a service role from within IAM. For more information, see Create a role to delegate permissions to an AWS service in the *IAM User Guide*.

> **Warning**
>
> Changing the permissions for a service role might break Amazon EBS functionality. Edit service roles only when Amazon EBS provides guidance to do so.

Service-linked roles for Amazon EBS

Supports service-linked roles: No

A service-linked role is a type of service role that is linked to an AWS service. The service can assume the role to perform an action on your behalf. Service-linked roles appear in your AWS account and are owned by the service. An IAM administrator can view, but not edit the permissions for service-linked roles.

For details about creating or managing service-linked roles, see AWS services that work with IAM. Find a service in the table that includes a `Yes` in the **Service-linked role** column. Choose the **Yes** link to view the service-linked role documentation for that service.

Example IAM policies for Amazon EBS

By default, users and roles don't have permission to create or modify Amazon EBS resources. They also can't perform tasks by using the AWS Management Console, AWS Command Line Interface (AWS CLI), or AWS API. To grant users permission to perform actions on the resources that they need, an IAM administrator can create IAM policies. The administrator can then add the IAM policies to roles, and users can assume the roles.

To learn how to create an IAM identity-based policy by using these example JSON policy documents, see Creating IAM policies in the *IAM User Guide*.

Topics

Policy best practices

Identity-based policies determine whether someone can create, access, or delete Amazon EBS resources in your account. These actions can incur costs for your AWS account. When you create or edit identity-based policies, follow these guidelines and recommendations:

- **Get started with AWS managed policies and move toward least-privilege permissions** – To get started granting permissions to your users and workloads, use the *AWS managed policies* that grant permissions for many common use cases. They are available in your AWS account. We recommend that you reduce permissions further by defining AWS customer managed policies that are specific to your use cases. For more information, see AWS managed policies or AWS managed policies for job functions in the *IAM User Guide*.
- **Apply least-privilege permissions** – When you set permissions with IAM policies, grant only the permissions required to perform a task. You do this by defining the actions that can be taken on

specific resources under specific conditions, also known as *least-privilege permissions*. For more information about using IAM to apply permissions, see Policies and permissions in IAM in the *IAM User Guide*.

- **Use conditions in IAM policies to further restrict access** – You can add a condition to your policies to limit access to actions and resources. For example, you can write a policy condition to specify that all requests must be sent using SSL. You can also use conditions to grant access to service actions if they are used through a specific AWS service, such as CloudFormation. For more information, see IAM JSON policy elements: Condition in the *IAM User Guide*.
- **Use IAM Access Analyzer to validate your IAM policies to ensure secure and functional permissions** – IAM Access Analyzer validates new and existing policies so that the policies adhere to the IAM policy language (JSON) and IAM best practices. IAM Access Analyzer provides more than 100 policy checks and actionable recommendations to help you author secure and functional policies. For more information, see Validate policies with IAM Access Analyzer in the *IAM User Guide*.
- **Require multi-factor authentication (MFA)** – If you have a scenario that requires IAM users or a root user in your AWS account, turn on MFA for additional security. To require MFA when API operations are called, add MFA conditions to your policies. For more information, see Secure API access with MFA in the *IAM User Guide*.

For more information about best practices in IAM, see Security best practices in IAM in the *IAM User Guide*.

Allow users to use the Amazon EBS console

To access the Amazon Elastic Block Store console, you must have a minimum set of permissions. These permissions must allow you to list and view details about the Amazon EBS resources in your AWS account. If you create an identity-based policy that is more restrictive than the minimum required permissions, the console won't function as intended for entities (users or roles) with that policy.

You don't need to allow minimum console permissions for users that are making calls only to the AWS CLI or the AWS API. Instead, allow access to only the actions that match the API operation that they're trying to perform.

To ensure that users and roles can still use the Amazon EBS console, also attach the Amazon EBS `ConsoleAccess` or `ReadOnly` AWS managed policy to the entities. For more information, see Adding permissions to a user in the *IAM User Guide*.

Allow users to view their own permissions

This example shows how you might create a policy that allows IAM users to view the inline and managed policies that are attached to their user identity. This policy includes permissions to complete this action on the console or programmatically using the AWS CLI or AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Allow users to work with volumes

Examples

- Example: Attach and detach volumes
- Example: Create a volume
- Example: Create a volume with tags
- Example: Work with volumes using the Amazon EC2 console

Example: Attach and detach volumes

When an API action requires a caller to specify multiple resources, you must create a policy statement that allows users to access all required resources. If you need to use a `Condition` element with one or more of these resources, you must create multiple statements as shown in this example.

The following policy allows users to attach volumes with the tag "`volume_user=`*iam-user-name*" to instances with the tag "`department=dev`", and to detach those volumes from those instances. If you attach this policy to an IAM group, the `aws:username` policy variable gives each user in the group permission to attach or detach volumes from the instances with a tag named `volume_user` that has their username as a value.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:instance/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/department": "dev"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
```

```
                "ec2:AttachVolume",
                "ec2:DetachVolume"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/volume_user": "${aws:username}"
                }
            }
        }
    ]
}
```

Example: Create a volume

The following policy allows users to use the CreateVolume API action. The user is allowed to create a volume only if the volume is encrypted and only if the volume size is less than 20 GiB.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateVolume"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:VolumeSize": "20"
                },
                "Bool": {
                    "ec2:Encrypted": "true"
                }
            }
        }
    ]
}
```

Example: Create a volume with tags

The following policy includes the `aws:RequestTag` condition key that requires users to tag any volumes they create with the tags `costcenter=115` and `stack=prod`. If users don't pass these specific tags, or if they don't specify tags at all, the request fails.

For resource-creating actions that apply tags, users must also have permissions to use the `CreateTags` action. The second statement uses the `ec2:CreateAction` condition key to allow users to create tags only in the context of `CreateVolume`. Users cannot tag existing volumes or any other resources.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCreateTaggedVolumes",
            "Effect": "Allow",
            "Action": "ec2:CreateVolume",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/costcenter": "115",
                    "aws:RequestTag/stack": "prod"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateVolume"
                }
            }
        }
    ]
```

```
}
```

The following policy allows users to create a volume without having to specify tags. The `CreateTags` action is only evaluated if tags are specified in the `CreateVolume` request. If users do specify tags, the tag must be `purpose=test`. No other tags are allowed in the request.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateVolume",
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
                "ec2:CreateTags"
            ],
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/purpose": "test",
                    "ec2:CreateAction": "CreateVolume"
                },
                "ForAllValues:StringEquals": {
                    "aws:TagKeys": "purpose"
                }
            }
        }
    ]
}
```

Example: Work with volumes using the Amazon EC2 console

The following policy grants users permission to view and create volumes, and attach and detach volumes to specific instances using the Amazon EC2 console.

Users can attach any volume to instances that have the tag "purpose=test", and also detach volumes from those instances. To attach a volume using the Amazon EC2 console, it is helpful for users to have permission to use the ec2:DescribeInstances action, as this allows them to select an instance from a pre-populated list in the **Attach Volume** dialog box. However, this also allows users to view all instances on the **Instances** page in the console, so you can omit this action.

In the first statement, the ec2:DescribeAvailabilityZones action is necessary to ensure that a user can select an Availability Zone when creating a volume.

Users cannot tag the volumes that they create (either during or after volume creation).

Allow users to work with snapshots

The following are example policies for both CreateSnapshot (point-in-time snapshot of an EBS volume) and CreateSnapshots (multi-volume snapshots).

Examples

- Example: Create a snapshot
- Example: Create snapshots
- Example: Create a snapshot with tags
- Example: Create multi-volume snapshots with tags
- Example: Copying snapshots
- Example: Modify permission settings for snapshots

Example: Create a snapshot

The following policy allows customers to use the CreateSnapshot API action. The customer can create snapshots only if the volume is encrypted and only if the volume size is less than 20 GiB.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*"
```

```
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "NumericLessThan": {
                    "ec2:VolumeSize": "20"
                },
                "Bool": {
                    "ec2:Encrypted": "true"
                }
            }
        }
    ]
}
```

Example: Create snapshots

The following policy allows customers to use the CreateSnapshots API action. The customer can create snapshots only if all of the volumes on the instance are type GP2.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshots",
         "Resource":[
"arn:aws:ec2:us-east-1::snapshot/*",
"arn:aws:ec2:*:*:instance/*"
   ]
      },
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshots",
         "Resource":"arn:aws:ec2:us-east-1:*:volume/*",
         "Condition":{
            "StringLikeIfExists":{
               "ec2:VolumeType":"gp2"
```

```
                }
        }

         }
    ]
}
```

Example: Create a snapshot with tags

The following policy includes the `aws:RequestTag` condition key that requires the customer to apply the tags `costcenter=115` and `stack=prod` to any new snapshot. If users don't pass these specific tags, or if they don't specify tags at all, the request fails.

For resource-creating actions that apply tags, customers must also have permissions to use the `CreateTags` action. The third statement uses the `ec2:CreateAction` condition key to allow customers to create tags only in the context of `CreateSnapshot`. Customers cannot tag existing volumes or any other resources.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*"
        },
        {
            "Sid": "AllowCreateTaggedSnapshots",
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshot",
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/costcenter": "115",
                    "aws:RequestTag/stack": "prod"
                }
            }
        },
        {
```

```
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "ec2:CreateAction": "CreateSnapshot"
                }
            }
        }
    ]
}
```

Example: Create multi-volume snapshots with tags

The following policy includes the `aws:RequestTag` condition key that requires the customer to apply the tags `costcenter=115` and `stack=prod` when creating a multi-volume snapshot set. If users don't pass these specific tags, or if they don't specify tags at all, the request fails.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshots",
         "Resource":[
"arn:aws:ec2:us-east-1::snapshot/*",
"arn:aws:ec2:*:*:instance/*",
"arn:aws:ec2:*:*:volume/*"

   ]
      },
      {
         "Sid":"AllowCreateTaggedSnapshots",
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshots",
         "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
         "Condition":{
            "StringEquals":{
               "aws:RequestTag/costcenter":"115",
```

```
                "aws:RequestTag/stack":"prod"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":"ec2:CreateTags",
         "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
         "Condition":{
            "StringEquals":{
               "ec2:CreateAction":"CreateSnapshots"
            }
         }
      }
   ]
}
```

The following policy allows customers to create a snapshot without having to specify tags. The `CreateTags` action is evaluated only if tags are specified in the `CreateSnapshot` or `CreateSnapshots` request. Tags can be omitted in the request. If a tag is specified, the tag must be `purpose=test`. No other tags are allowed in the request.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect":"Allow",
         "Action":"ec2:CreateSnapshot",
         "Resource":"*"
      },
      {
         "Effect":"Allow",
         "Action":"ec2:CreateTags",
         "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
         "Condition":{
            "StringEquals":{
               "aws:RequestTag/purpose":"test",
               "ec2:CreateAction":"CreateSnapshot"
            },
```

```
                "ForAllValues:StringEquals":{
                    "aws:TagKeys":"purpose"
                }
            }
        }
    ]
}
```

The following policy allows customers to create multi-volume snapshot sets without having to specify tags. The `CreateTags` action is evaluated only if tags are specified in the `CreateSnapshot` or `CreateSnapshots` request. Tags can be omitted in the request. If a tag is specified, the tag must be `purpose=test`. No other tags are allowed in the request.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect":"Allow",
            "Action":"ec2:CreateSnapshots",
            "Resource":"*"
        },
        {
            "Effect":"Allow",
            "Action":"ec2:CreateTags",
            "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
            "Condition":{
                "StringEquals":{
                    "aws:RequestTag/purpose":"test",
                    "ec2:CreateAction":"CreateSnapshots"
                },
                "ForAllValues:StringEquals":{
                    "aws:TagKeys":"purpose"
                }
            }
        }
    ]
}
```

The following policy allows snapshots to be created only if the source volume is tagged with `User:username` for the customer, and the snapshot itself is tagged with `Environment:Dev` and `User:username`. The customer can add additional tags to the snapshot.

The following policy for `CreateSnapshots` allows snapshots to be created only if the source volume is tagged with `User:username` for the customer, and the snapshot itself is tagged with `Environment:Dev` and `User:username`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshots",
            "Resource": "arn:aws:ec2:us-east-1:*:instance/*"
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshots",
            "Resource": "arn:aws:ec2:us-east-1:111122223333:volume/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/User": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateSnapshots",
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/Environment": "Dev",
                    "aws:RequestTag/User": "${aws:username}"
                }
            }
        },
        {
            "Effect": "Allow",
            "Action": "ec2:CreateTags",
```

```
            "Resource": "arn:aws:ec2:us-east-1::snapshot/*"
        }
    ]
}
```

The following policy allows deletion of a snapshot only if the snapshot is tagged with User:*username* for the customer.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
            {
         "Effect":"Allow",
         "Action":"ec2:DeleteSnapshot",
         "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
         "Condition":{
            "StringEquals":{
               "aws:ResourceTag/User":"${aws:username}"
            }
         }
      }
   ]
}
```

The following policy allows a customer to create a snapshot but denies the action if the snapshot being created has a tag key `value=stack`.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
         "Effect":"Allow",
         "Action":[
            "ec2:CreateSnapshot",
            "ec2:CreateTags"
```

```
        ],
        "Resource":"*"
      },
      {
        "Effect":"Deny",
        "Action":"ec2:CreateSnapshot",
        "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
        "Condition":{
          "ForAnyValue:StringEquals":{
            "aws:TagKeys":"stack"
          }
        }
      }
   ]
}
```

The following policy allows a customer to create snapshots but denies the action if the snapshots being created have a tag key `value=stack`.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
      {
        "Effect":"Allow",
        "Action":[
          "ec2:CreateSnapshots",
          "ec2:CreateTags"
        ],
        "Resource":"*"
      },
      {
        "Effect":"Deny",
        "Action":"ec2:CreateSnapshots",
        "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
        "Condition":{
          "ForAnyValue:StringEquals":{
            "aws:TagKeys":"stack"
          }
        }
```

```
        }
    ]
}
```

The following policy allows you to combine multiple actions into a single policy. You can only create a snapshot (in the context of `CreateSnapshots`) when the snapshot is created in Region `us-east-1`. You can only create snapshots (in the context of `CreateSnapshots`) when the snapshots are being created in the Region `us-east-1` and when the instance type is `t2*`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
           {
            "Effect":"Allow",
            "Action":[
               "ec2:CreateSnapshots",
               "ec2:CreateSnapshot",
               "ec2:CreateTags"
            ],
            "Resource": [
               "arn:aws:ec2:*:*:instance/*",
               "arn:aws:ec2:*:*:snapshot/*",
               "arn:aws:ec2:*:*:volume/*"
            ],
            "Condition":{
               "StringEqualsIgnoreCase": {
                 "ec2:Region": "us-east-1"
               },
               "StringLikeIfExists": {
                 "ec2:InstanceType": ["t2.*"]
               }
            }
        }
    ]
}
```

Example: Copying snapshots

Resource-level permissions specified for the **CopySnapshot** action apply to both the new snapshot and the source snapshot.

The following example policy allows principals to copy snapshots only if the new snapshot is created with tag key of `purpose` and a tag value of `production`(`purpose=production`).

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "AllowCopySnapshotWithTags",
            "Effect": "Allow",
            "Action": "ec2:CopySnapshot",
            "Resource": "arn:aws:ec2:*:111122223333:snapshot/*",
            "Condition": {
                "StringEquals": {
                    "aws:RequestTag/purpose": "production"
                }
            }
        }
    ]
}
```

The following example policy allows principals to copy snapshots only if the source snapshot is owned by AWS account `123456789012`.

The following example policy allows principals to copy snapshots only if the source snapshot is created with tag key of `CSISnapshotName`.

```
{
    "Effect": "Allow",
    "Action": "ec2:CopySnapshot",
    "Resource": "arn:aws:ec2:*::snapshot/${*}",
    "Condition": {
        "StringLike": {
            "aws:RequestTag/CSISnapshotName": "*"
```

```
        }
    }
},

{
    "Effect": "Allow",
    "Action": "ec2:CopySnapshot",
    "Resource": "arn:aws:ec2:*::snapshot/snap-*"
}
```

Example: Modify permission settings for snapshots

The following policy allows modification of a snapshot only if the snapshot is tagged with `User:`*`username`*, where *`username`* is the customer's AWS account user name. The request fails if this condition is not met.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
            {
         "Effect":"Allow",
         "Action":"ec2:ModifySnapshotAttribute",
         "Resource":"arn:aws:ec2:us-east-1::snapshot/*",
         "Condition":{
            "StringEquals":{
               "aws:ResourceTag/user-name":"${aws:username}"
            }
         }
      }
   ]
}
```

Troubleshoot Amazon EBS authorization issues

Use the following information to help you diagnose and fix common issues that you might encounter when working with Amazon EBS and IAM.

Issues

I am not authorized to perform an action in Amazon EBS

If the AWS Management Console tells you that you're not authorized to perform an action, then you must contact your administrator for assistance. Your administrator is the person that provided you with your sign-in credentials.

The following example error occurs when the `mateojackson` IAM user tries to use the console to view details about a volume but does not have `ec2:DescribeVolumes` permissions.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ec2:DescribeVolumes on resource: volume-id
```

In this case, Mateo asks his AWS administrator to allow him to describe the volume.

I am not authorized to perform iam:PassRole

If you receive an error that you're not authorized to perform the `iam:PassRole` action, your policies must be updated to allow you to pass a role to Amazon EBS.

Some AWS services allow you to pass an existing role to that service instead of creating a new service role or service-linked role. To do this, you must have permissions to pass the role to the service.

The following example error occurs when an IAM user named `marymajor` tries to use the console to perform an action in Amazon EBS. However, the action requires the service to have permissions that are granted by a service role. Mary does not have permissions to pass the role to the service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

In this case, Mary's policies must be updated to allow her to perform the `iam:PassRole` action.

If you need help, contact your AWS administrator. Your administrator is the person who provided you with your sign-in credentials.

I want to allow people outside of my AWS account to access my Amazon EBS resources

You can create a role that users in other accounts or people outside of your organization can use to access your resources. You can specify who is trusted to assume the role. For services that support resource-based policies or access control lists (ACLs), you can use those policies to grant people access to your resources.

To learn more, consult the following:

- To learn whether Amazon EBS supports these features, see How Amazon EBS works with IAM.
- To learn how to provide access to your resources across AWS accounts that you own, see Providing access to an IAM user in another AWS account that you own in the *IAM User Guide*.
- To learn how to provide access to your resources to third-party AWS accounts, see Providing access to AWS accounts owned by third parties in the *IAM User Guide*.
- To learn how to provide access through identity federation, see Providing access to externally authenticated users (identity federation) in the *IAM User Guide*.
- To learn the difference between using roles and resource-based policies for cross-account access, see Cross account resource access in IAM in the *IAM User Guide*.

Compliance validation for Amazon EBS

To learn whether an AWS service is within the scope of specific compliance programs, see AWS services in Scope by Compliance Program and choose the compliance program that you are interested in. For general information, see AWS Compliance Programs.

You can download third-party audit reports using AWS Artifact. For more information, see Downloading Reports in AWS Artifact.

Your compliance responsibility when using AWS services is determined by the sensitivity of your data, your company's compliance objectives, and applicable laws and regulations. For more information about your compliance responsibility when using AWS services, see AWS Security Documentation.

Data resiliency in Amazon EBS

The AWS global infrastructure is built around AWS Regions and Availability Zones. AWS Regions provide multiple physically separated and isolated Availability Zones, which are connected with low-latency, high-throughput, and highly redundant networking. With Availability Zones, you can design and operate applications and databases that automatically fail over between zones without interruption. Availability Zones are more highly available, fault tolerant, and scalable than traditional single or multiple data center infrastructures.

For more information about AWS Regions and Availability Zones, see AWS Global Infrastructure.

In addition to the AWS global infrastructure, Amazon EBS offers several features to help support your data resiliency and backup needs.

- Automating EBS snapshots using Amazon Data Lifecycle Manager
- Copying EBS snapshots across Regions

Monitoring tools for Amazon EBS

Monitoring is an important part of maintaining the reliability, availability, and performance of Amazon Elastic Block Store and your other AWS solutions. AWS provides the following monitoring tools to watch Amazon EBS, report when something is wrong, and take automatic actions when appropriate:

- **AWS CloudTrail** captures API calls and related events made by or on behalf of your AWS account and delivers the log files to an Amazon S3 bucket that you specify. You can identify which users and accounts called AWS, the source IP address from which the calls were made, and when the calls occurred. The APIs to manage your EBS volumes and snapshots are part of the Amazon EC2 API. For more information about CloudTrail and the Amazon EC2 API, see Log Amazon EC2 API calls using AWS CloudTrail in the *Amazon EC2 User Guide*.
- **Amazon CloudWatch** monitors your AWS resources and the applications you run on AWS in real time. You can collect and track metrics, create customized dashboards, and set alarms that notify you or take actions when a specified metric reaches a threshold that you specify. For example, you can have CloudWatch track CPU usage or other metrics of your Amazon EC2 instances and automatically launch new instances when needed. For more information, see the section called "Amazon CloudWatch".
- **Amazon EventBridge** can be used to automate your AWS services and respond automatically to system events, such as application availability issues or resource changes. Events from AWS services are delivered to EventBridge in near real time. You can write simple rules to indicate which events are of interest to you and which automated actions to take when an event matches a rule. For more information, see the section called "Amazon EventBridge".
- **Amazon EBS detailed performance statistics** provide real-time I/O performance statistics for Amazon EBS volumes attached to Nitro-based Amazon EC2 instances. For more information, Amazon EBS detailed performance statistics.
- **Amazon GuardDuty** helps detect potentially malicious activity in your EC2 instances. GuardDuty Malware Protection for EC2 scans the EBS volumes attached to your EC2 instances. For more information, see the section called "Amazon GuardDuty".

Amazon CloudWatch metrics for Amazon EBS

Amazon CloudWatch metrics are statistical data that you can use to view, analyze, and set alarms on the operational behavior of your volumes.

Data is available automatically in 1-minute periods at no charge.

When you get data from CloudWatch, you can include a `Period` request parameter to specify the granularity of the returned data. This is different than the period that we use when we collect the data (1-minute periods). We recommend that you specify a period in your request that is equal to or greater than the collection period to ensure that the returned data is valid.

You can get the data using either the CloudWatch API or the Amazon EC2 console. The console takes the raw data from the CloudWatch API and displays a series of graphs based on the data. Depending on your needs, you might prefer to use either the data from the API or the graphs in the console.

Topics

Metrics for Amazon EBS volumes

The `AWS/EBS` namespace includes the following metrics for EBS volumes that are attached to all instance types. All Amazon EBS volume types automatically send 1-minute metrics to CloudWatch, but only when the volume is attached to an instance.

To get information about the available disk space from the operating system on an instance, see View free disk space.

> **Note**
>
> Some metrics have differences on instances that are built on the Nitro System. For a list of these instance types, see Instances built on the Nitro System.

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeAvgIOPS`	**Note** Supported for all EBS volume types attached to Nitro instances. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. The average read and write IOPS driven to the volume in a minute. If no operations were driven to the volume within the last minute, then value for the metric is zero (`0`). For more information, see [Monitor I/O characteristics using CloudWatch](). For Multi-Attach enabled volumes, use the `InstanceId` dimension to view average IOPS for a specific volume-instance attachement.	Ops/s	`VolumeId` | `InstanceId`	• `Sum` • `Average` • `Minimum` | `Maximum`

Metric	Description	Units	Dimensions	Meaningful statistics
VolumeAvgThroughput	**Note** Supported for all EBS volume types attached to Nitro instances. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. The average read and write throughput driven to the volume in a minute. If no operations were driven to the volume within the last minute, then value for the metric is zero (0). For more information, see Monitor I/O characteristics using CloudWatch. For Multi-Attach enabled volumes, use the InstanceId dimension to view average throughput for a specific volume-instance attachement..	KiB/s	VolumeId | InstanceId	• Sum • Average • Minimum | Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
VolumeAvgReadLatency	**Note** Supported for all volume types attached to Nitro instances. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. The average time taken to complete read operations in a minute. Use this metric to monitor the average I/O latency of the EBS volumes attached to your Amazon EC2 instances. The average is calculated based on I/O operations that completed in the last minute. If no operations completed within the last minute, then value for the metric is zero. For Multi-Attach enabled volumes, use the InstanceId	Milliseconds	VolumeId | InstanceId	Minimum | Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
	dimension to view average latency for a specific volume-instance attachement.			

Metric	Description	Units	Dimensions	Meaningful statistics
VolumeAvgWriteLatency	**Note** Supported for all volume types attached to Nitro instances. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. The average time taken to complete write operations in a minute. Use this metric to monitor the average I/O latency of the EBS volumes attached to your Amazon EC2 instances. The average is calculated based on I/O operations that completed in the last minute. If no operations completed within the last minute, then value for the metric is zero. For Multi-Attach enabled volumes, use the InstanceId	Milliseconds	VolumeId | InstanceId	Minimum | Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
	dimension to view average latency for a specific volume-instance attachement.			

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeIOPSExceededCheck`	**Note** Supported for all volume types, except magnetic (`standard`), attached to Nitro instances. Not supported with Multi-Attach enabled volumes. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. Reports whether an application consistently attempted to drive IOPS that exceeds the volume's provisioned IOPS performance within the last minute. This metric can be either `0` (provisioned IOPS not exceeded) or 1 (provisioned IOPS exceeded). For more information, see [Monitor I/O characteristics using CloudWatch]().	None	`VolumeId` | `InstanceId`	• `Sum` • `Average` • `Minimum` | `Maximum`

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeThroughputExceededCheck`	**Note** Supported for all volume types, except magnetic (`standard`), attached to Nitro instances. Not supported with Multi-Attach enabled volumes. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. Reports whether an application consistently attempted to drive throughput that exceeds the volume's provisioned throughput performance within the last minute. This metric can be either `0` (provisioned throughput not exceeded) or 1 (provisioned throughput exceeded).For more information, see Monitor	None	`VolumeId` | `InstanceId`	• `Sum` • `Average` • `Minimum` | `Maximum`

Metric	Description	Units	Dimensions	Meaningful statistics
	[I/O characteristics using CloudWatch]().			
`VolumeReadBytes`	Provides information on the read operations in a specified period of time. • The `Sum` statistic reports the total number of bytes transferred during the period. • The `Average` statistic reports the average number of bytes read over the specified period. • The `SampleCount` statistic represents the number of data points used in the statistical calculation. **Note** For Xen instances, data is reported only when there is read activity on the volume.	Bytes	`VolumeId`	• `Average` • `Sum` • `SampleCount` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeWriteBytes`	Provides information on the write operations in a specified period of time • The `Sum` statistic reports the total number of bytes transferred during the period. • The `Average` statistic reports the average number of bytes written over the specified period. • The `SampleCount` statistic represents the number of data points used in the statistical calculation. **Note** For Xen instances, data is reported only when there is write activity on the volume.	Bytes	`VolumeId`	• `Average` • `Sum` • `SampleCount` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeReadOps`	The total number of read operations in a specified period of time. Read operations are counted on completion. To calculate the average read operations per second (read IOPS) for the period, divide the total read operations in the period by the number of seconds in that period.	Count	`VolumeId`	• `Average` • `Sum` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances
`VolumeWriteOps`	The total number of write operations in a specified period of time. Write operations are counted on completion. To calculate the average write operations per second (write IOPS) for the period, divide the total write operations in the period by the number of seconds in that period.	Count	`VolumeId`	• `Average` • `Sum` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
VolumeTotalReadTime	**Note** Not supported with Multi-Attach enabled volumes. For Xen instances, data is reported only when there is read activity on the volume. The total number of seconds spent by all read operations that completed in a specified period of time. If multiple requests are submitted at the same time, this total could be greater than the length of the period. For example, for a period of 1 minutes (60 seconds): if 150 operations completed during that period, and each operation took 1 second, the value would be 150 seconds.	Seconds	VolumeId	• Average — not relevant for volumes attached to Nitro-based instances • Sum • Minimum | Maximum — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeTotalWriteTime`	**Note** Not supported with Multi-Attach enabled volumes. For Xen instances, data is reported only when there is write activity on the volume. The total number of seconds spent by all write operations that completed in a specified period of time. If multiple requests are submitted at the same time, this total could be greater than the length of the period. For example, for a period of 1 minute (60 seconds): if 150 operations completed during that period, and each operation took 1 second, the value would be 150 seconds.	Seconds	`VolumeId`	• `Average` — not relevant for volumes attached to Nitro-based instances • `Sum` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeIdleTime`	**Note** Not supported with Multi-Attach enabled volumes. The total number of seconds in a specified period of time when no read or write operations were submitted.	Seconds	`VolumeId`	• `Average` — not relevant for volumes attached to Nitro-based instances • `Sum` • `Minimum` | `Maximum` — only for volumes attached to Nitro-based instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeQueueLength`	The number of read and write operation requests waiting to be completed in a specified period of time.	Count	`VolumeId`	• `Average` • `Sum` — not relevant for volumes attached to Nitro instances • `Minimum` | `Maximum` — only for volumes attached to Nitro instances

Metric	Description	Units	Dimensions	Meaningful statistics
`VolumeStalledIOCheck`	**Note** For Nitro instances only. Not published for volumes attached to Amazon ECS and AWS Fargate tasks. Reports whether a volume has passed or failed a *stalled IO check* in the last minute.This metric can be either `0` (passed) or 1 (failed). For more information, see [Monitor I/O characteristics using CloudWatch]().	None	`VolumeId` | `InstanceId`	• Sum • Average • Minimum • Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
VolumeThroughputPercentage	**Note** Provisioned IOPS SSD volumes only. Not supported with Multi-Attach enabled volumes. The percentage of I/O operations per second (IOPS) delivered of the total IOPS provisioned for an Amazon EBS volume. Provisioned IOPS SSD volumes deliver their provisioned performance 99.9 percent of the time. During a write, if there are no other pending I/O requests in a minute, the metric value will be 100 percent. Also, a volume's I/O performance may become degraded temporarily due to an action you have taken (for example, creating a snapshot of a volume during peak usage, running the volume on	Percent	VolumeId	• Average • Minimum | Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
	a non-EBS-optimized instance, or accessing data on the volume for the first time).			
`VolumeConsumedReadWriteOps`	**Note** Provisioned IOPS SSD volumes only. The total amount of read and write operations (normalized to 256K capacity units) consumed in a specified period of time. I/O operations that are smaller than 256K each count as 1 consumed IOPS. I/O operations that are larger than 256K are counted in 256K capacity units. For example, a 1024K I/O would count as 4 consumed IOPS.	Count	`VolumeId`	• `Average` • `Sum` • `Minimum` | `Maximum`

Metric	Description	Units	Dimensions	Meaningful statistics
BurstBalance	**Note** gp2, st1, and sc1 volumes only. Provides information about the percentage of I/O credits (for gp2) or throughput credits (for st1 and sc1) remaining in the burst bucket. Data is reported to CloudWatch only when the volume is active. If the volume is not attached, no data is reported. If the baseline performance of the volume exceeds the maximum burst performance, credits are never spent. If the volume is attached to an instance built on the Nitro System, the burst balance is not reported. For other instances, the reported burst balance is 100%. For more information, see	Percent	VolumeId	• Average • Sum — not relevant for volumes attached to Nitro instances. • Minimum | Maximum

Metric	Description	Units	Dimensions	Meaningful statistics
	gp2 volume performance.			

Metrics for Amazon EBS snapshots

The `AWS/EBS` namespace includes the following metrics for Amazon EBS snapshots.

Metric	Description	Units	Dimensions	Meaningful statistics
`SnapshotCopyBytesTransferred`	The amount of snapshot data copied to an AWS Region.	Bytes	`sourceRegion`	Sum

Metrics for Nitro instances

The `AWS/EC2` namespace includes additional Amazon EBS metrics for volumes that are attached to Nitro-based instances that are not bare metal instances.

Metric	Description	Unit	Meaningful statistics
`InstanceEBSIOPSExceededCheck`	Reports whether an application attempted to drive IOPS that exceeds the maximum EBS IOPS limits for the instance within the last minute. This metric can be either `0` (IOPS not exceeded) or 1 (IOPS exceeded).	None	• Sum • Average • Minimum • Maximum
`InstanceEBSThroughputExceededCheck`	Reports whether an application attempted to drive throughput that exceeds the maximum EBS throughput limits for the instance within the last minute. This metric can be either `0`	None	• Sum • Average • Minimum

Metric	Description	Unit	Meaningful statistics
	(throughput not exceeded) or 1 (throughput exceeded).		• Maximum
`EBSReadOps`	Completed read operations from all Amazon EBS volumes attached to the instance in a specified period of time. To calculate the average read I/O operations per second (Read IOPS) for the period, divide the total operations in the period by the number of seconds in that period. If you are using basic (5-minute) monitoring, you can divide this number by 300 to calculate the Read IOPS. If you have detailed (1-minute) monitoring, divide it by 60. You can also use the CloudWatch metric math function `DIFF_TIME` to find the operations per second. For example, if you have graphed `EBSReadOps` in CloudWatch as `m1`, the metric math formula `m1/(DIFF_TIME(m1))` returns the metric in operations/second. For more information about `DIFF_TIME` and other metric math functions, see [Use metric math] in the *Amazon CloudWatch User Guide*.	Count	• Sum • Average • Minimum • Maximum

Metric	Description	Unit	Meaningful statistics
`EBSWriteOps`	Completed write operations to all EBS volumes attached to the instance in a specified period of time. To calculate the average write I/O operations per second (Write IOPS) for the period, divide the total operations in the period by the number of seconds in that period. If you are using basic (5-minute) monitoring, you can divide this number by 300 to calculate the Write IOPS. If you have detailed (1-minute) monitoring, divide it by 60. You can also use the CloudWatch metric math function `DIFF_TIME` to find the operations per second. For example, if you have graphed `EBSWriteOps` in CloudWatch as `m1`, the metric math formula `m1/(DIFF_TIME(m1))` returns the metric in operations/second. For more information about `DIFF_TIME` and other metric math functions, see [Use metric math](#) in the *Amazon CloudWatch User Guide*.	Count	• Sum • Average • Minimum • Maximum

Metric	Description	Unit	Meaningful statistics
`EBSReadBytes`	Bytes read from all EBS volumes attached to the instance in a specified period of time. The number reported is the number of bytes read during the period. If you are using basic (5-minute) monitoring, you can divide this number by 300 to find Read Bytes/second. If you have detailed (1-minute) monitoring, divide it by 60. You can also use the CloudWatch metric math function `DIFF_TIME` to find the bytes per second. For example, if you have graphed `EBSReadBytes` in CloudWatch as `m1`, the metric math formula `m1/(DIFF_TIME(m1))` returns the metric in bytes/second. For more information about `DIFF_TIME` and other metric math functions, see [Use metric math]() in the *Amazon CloudWatch User Guide*.	Bytes	• Sum • Average • Minimum • Maximum

Metric	Description	Unit	Meaningful statistics
EBSWriteBytes	Bytes written to all EBS volumes attached to the instance in a specified period of time. The number reported is the number of bytes written during the period. If you are using basic (5-minute) monitoring, you can divide this number by 300 to find Write Bytes/second. If you have detailed (1-minute) monitoring, divide it by 60. You can also use the CloudWatch metric math function `DIFF_TIME` to find the bytes per second. For example, if you have graphed `EBSWriteBytes` in CloudWatch as `m1`, the metric math formula `m1/(DIFF_TIME(m1))` returns the metric in bytes/second. For more information about `DIFF_TIME` and other metric math functions, see [Use metric math](#) in the *Amazon CloudWatch User Guide*.	Bytes	• Sum • Average • Minimum • Maximum
EBSIOBalance%	Provides information about the percentage of I/O credits remaining in the burst bucket. This metric is available for basic monitoring only. This metric is available only for some `*.4xlarge` instance sizes and smaller that burst to their maximum performance for only 30 minutes at least once every 24 hours. For more information, see [EBS optimized by default](#). The `Sum` statistic is not applicable to this metric.	Percent	• Minimum • Maximum

Metric	Description	Unit	Meaningful statistics
EBSByteBalance%	Provides information about the percentage of throughput credits remaining in the burst bucket. This metric is available for basic monitoring only. This metric is available only for some *.4xlarge instance sizes and smaller that burst to their maximum performance for only 30 minutes at least once every 24 hours. For more information, see EBS optimized by default. The Sum statistic is not applicable to this metric.	Percent	• Minimum • Maximum

Metrics for fast snapshot restore

AWS/EBS namespace includes the following metrics for fast snapshot restore.

Metric	Description	Units	Dimensions	Meaningful statistics
FastSnapshotRestoreCreditsBucketSize	The maximum number of volume create credits that can be accumulated. This metric is reported per snapshot per Availability Zone.	None	SnapshotId | AvailabilityZone	• Average • Minimum | Maximum **Note** The most meaningful statistic is Average. The results for the Minimum and Maximum statistics are the same as

Metric	Description	Units	Dimensions	Meaningful statistics
				for Average and could be used instead.
FastSnapshotRestoreCreditsBalance	The number of volume create credits available. This metric is reported per snapshot per Availability Zone.	None	SnapshotId | AvailabilityZone	• Average • Minimum | Maximum **Note** The most meaningful statistic is Average. The results for the Minimum and Maximum statistics are the same as for Average and could be used instead.

Amazon EC2 console graphs

After you create a volume, you can view the volume's monitoring graphs in the Amazon EC2 console. Select a volume on the **Volumes** page in the console and choose **Monitoring**. The following table lists the graphs that are displayed. The column on the right describes how the raw data metrics from the CloudWatch API are used to produce each graph. The period for all the graphs is 5 minutes.

Graph	Description using raw metrics
Read throughput (KiB/s)	`Sum(VolumeReadBytes) / Period / 1024`
Write throughput (KiB/s)	`Sum(VolumeWriteBytes) / Period / 1024`
Read operations (Ops/s)	`Sum(VolumeReadOps) / Period`
Write operations (Ops/s)	`Sum(VolumeWriteOps) / Period`
Average queue length (Operations)	`Avg(VolumeQueueLength)`
Time spent idle (%)	`Sum(VolumeIdleTime) / Period × 100`

Average read size (KiB/op)	`Avg(VolumeReadBytes) / 1024`
	
	For Nitro-based instances, the following formula derives Average Read Size using CloudWatch Metric Math:
	
	`(Sum(VolumeReadBytes) / Sum(VolumeReadOps)) / 1024`
	
	The `VolumeReadBytes` and `VolumeReadOps` metrics are available in the EBS CloudWatch console.

Average write size (KiB/op)	`Avg(VolumeWriteBytes) / 1024`
	
	For Nitro-based instances, the following formula derives Average Write Size using CloudWatch Metric Math:
	
	`(Sum(VolumeWriteBytes) / Sum(VolumeWriteOps)) / 1024`
	
	The `VolumeWriteBytes` and `VolumeWriteOps` metrics are available in the EBS CloudWatch console.

Average read latency (ms/op)	`Avg(VolumeTotalReadTime) × 1000` For Nitro-based instances, the following formula derives Average Read Latency using CloudWatch Metric Math:

Graph	Description using raw metrics
	(Sum(VolumeTotalReadTime) / Sum(VolumeReadOps)) × 1000 The VolumeTotalReadTime and VolumeReadOps metrics are available in the EBS CloudWatch console.
Average write latency (ms/op)	Avg(VolumeTotalWriteTime) × 1000 For Nitro-based instances, the following formula derives Average Write Latency using CloudWatch Metric Math: (Sum(VolumeTotalWriteTime) / Sum(VolumeWriteOps)) * 1000 The VolumeTotalWriteTime and VolumeWriteOps metrics are available in the EBS CloudWatch console.

For the average latency graphs and average size graphs, the average is calculated over the total number of operations (read or write, whichever is applicable to the graph) that completed during the period.

Amazon EventBridge events for Amazon EBS

Amazon EBS sends events to Amazon EventBridge for actions performed on volumes and snapshots. With EventBridge, you can establish rules that trigger programmatic actions in response to these events. For example, you can create a rule that sends a notification to your email when a snapshot is enabled for fast snapshot restore.

Events in EventBridge are represented as JSON objects. The fields that are unique to the event are contained in the "detail" section of the JSON object. The "event" field contains the event name. The "result" field contains the completed status of the action that triggered the event. For more information, see Amazon EventBridge event patterns in the *Amazon EventBridge User Guide*.

For more information, see What Is Amazon EventBridge? in the *Amazon EventBridge User Guide*.

Events

- EBS volume events

- EBS volume initialization event
- EBS volume modification events
- EBS snapshot events
- EBS Snapshots Archive events
- EBS fast snapshot restore events
- Using AWS Lambda to handle EventBridge events

EBS volume events

Amazon EBS sends events to EventBridge when the following volume events occur.

Events

- Create volume (createVolume)
- Delete volume (deleteVolume)
- Volume attach or reattach (attachVolume, reattachVolume)
- Detach volume (detachVolume)

Create volume (createVolume)

The `createVolume` event is sent to your AWS account when an action to create a volume completes. However, it is not saved, logged, or archived. This event can have a result of either `available` or `failed`. Creation will fail if an invalid AWS KMS key was provided, as shown in the examples below.

Event data

The listing below is an example of a JSON object emitted by EBS for a successful `createVolume` event.

```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
   "detail-type": "EBS Volume Notification",
   "source": "aws.ec2",
   "account": "012345678901",
```

```
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": [
       "arn:aws:ec2:us-east-1:012345678901:volume/vol-01234567"
    ],
    "detail": {
       "result": "available",
       "cause": "",
       "event": "createVolume",
       "request-id": "01234567-0123-0123-0123-0123456789ab"
    }
}
```

The listing below is an example of a JSON object emitted by EBS after a failed `createVolume` event. The cause for the failure was a disabled KMS key.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-0123456789ab",
  "detail-type": "EBS Volume Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "sa-east-1",
  "resources": [
    "arn:aws:ec2:sa-east-1:0123456789ab:volume/vol-01234567",
  ],
  "detail": {
    "event": "createVolume",
    "result": "failed",
    "cause": "arn:aws:kms:sa-
east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab is disabled.",
    "request-id": "01234567-0123-0123-0123-0123456789ab",
  }
}
```

The following is an example of a JSON object that is emitted by EBS after a failed `createVolume` event. The cause for the failure was a KMS key pending import.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-0123456789ab",
```

```
  "detail-type": "EBS Volume Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "sa-east-1",
  "resources": [
    "arn:aws:ec2:sa-east-1:0123456789ab:volume/vol-01234567",
  ],
  "detail": {
    "event": "createVolume",
    "result": "failed",
    "cause": "arn:aws:kms:sa-
east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab is pending import.",
    "request-id": "01234567-0123-0123-0123-0123456789ab",
  }
}
```


Delete volume (deleteVolume)

The `deleteVolume` event is sent to your AWS account when an action to delete a volume completes. However, it is not saved, logged, or archived. This event has the result `deleted`. If the deletion does not complete, the event is never sent.

Event data

The listing below is an example of a JSON object emitted by EBS for a successful `deleteVolume` event.


```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
   "detail-type": "EBS Volume Notification",
   "source": "aws.ec2",
   "account": "012345678901",
   "time": "yyyy-mm-ddThh:mm:ssZ",
   "region": "us-east-1",
   "resources": [
      "arn:aws:ec2:us-east-1:012345678901:volume/vol-01234567"
   ],
   "detail": {
      "result": "deleted",
      "cause": "",
      "event": "deleteVolume",
```

```
        "request-id": "01234567-0123-0123-0123-0123456789ab"
    }
}
```

Volume attach or reattach (attachVolume, reattachVolume)

The `attachVolume` or `reattachVolume` event is sent to your AWS account when a volume is attached or reattached to an instance. However, it is not saved, logged, or archived. If you use a KMS key to encrypt an EBS volume and the KMS key becomes invalid, EBS will emit an event if that KMS key is later used to attach or reattach to an instance, as shown in the examples below.

Event data

The listing below is an example of a JSON object emitted by EBS after a failed `attachVolume` event. The cause for the failure was a KMS key pending deletion.

> **Note**
>
> AWS may attempt to reattach to a volume following routine server maintenance.


```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-0123456789ab",
  "detail-type": "EBS Volume Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
  "arn:aws:ec2:us-east-1:0123456789ab:volume/vol-01234567",
  "arn:aws:kms:us-east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab"
  ],
  "detail": {
    "event": "attachVolume",
    "result": "failed",
    "cause": "arn:aws:kms:us-
east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab is pending deletion.",
    "request-id": ""
  }
}
```

The listing below is an example of a JSON object emitted by EBS after a failed `reattachVolume` event. The cause for the failure was a KMS key pending deletion.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-0123456789ab",
  "detail-type": "EBS Volume Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
  "arn:aws:ec2:us-east-1:0123456789ab:volume/vol-01234567",
  "arn:aws:kms:us-east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab"
  ],
  "detail": {
    "event": "reattachVolume",
    "result": "failed",
    "cause": "arn:aws:kms:us-
east-1:0123456789ab:key/01234567-0123-0123-0123-0123456789ab is pending deletion.",
    "request-id": ""
  }
}
```

Detach volume (detachVolume)

The `detachVolume` event is sent to your AWS account when a volume is detached from an Amazon EC2 instance.

Event data

The following is an example of a successful `detachVolume` event.

```
{
  "version":"0",
  "id":"2ec37298-1234-e436-70fc-c96b1example",
  "detail-type":"AWS API Call via CloudTrail",
  "source":"aws.ec2",
  "account":"123456789012",
  "time":"2024-03-18T16:35:52Z",
  "region":"us-east-1",
  "resources":[],
  "detail":
```

```
  {
    "eventVersion":"1.09",
    "userIdentity":
    {
      "type":"IAMUser",
      "principalId":"AIDAJT12345SQ2EXAMPLE",
      "arn":"arn:aws:iam::123456789012:user/administrator",
      "accountId":"123456789012",
      "accessKeyId":"AKIAJ67890A6EXAMPLE",
      "userName":"administrator"
    },
    "eventTime":"2024-03-18T16:35:52Z",
    "eventSource":"ec2.amazonaws.com",
    "eventName":"DetachVolume",
    "awsRegion":"us-east-1",
    "sourceIPAddress":"12.12.123.12",
    "userAgent":"aws-cli/2.7.12 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ec2.detach-volume",
    "requestParameters":
    {
      "volumeId":"vol-072577c46bexample",
      "force":false
    },
    "responseElements":
    {
      "requestId":"1234513a-6292-49ea-83f8-85e95example",
      "volumeId":"vol-072577c46bexample",
      "instanceId":"i-0217f7eb3dexample",
      "device":"/dev/sdb",
      "status":"detaching",
      "attachTime":1710776815000
    },
    "requestID":"1234513a-6292-49ea-83f8-85e95example",
    "eventID":"1234551d-a15a-43eb-9e69-c983aexample",
    "readOnly":false,
    "eventType":"AwsApiCall",
    "managementEvent":true,
    "recipientAccountId":"123456789012",
    "eventCategory":"Management",
    "tlsDetails":
    {
      "tlsVersion":"TLSv1.3",
      "cipherSuite":"TLS_AES_128_GCM_SHA256",
      "clientProvidedHostHeader":"ec2.us-east-1.amazonaws.com"
```

```
    }
  }
}
```

EBS volume initialization event

When you create an Amazon EBS volume from a snapshot and use the default volume initialization rate or an Amazon EBS Provisioned Rate for Volume Initialization, the `initializeVolume` event is sent to your AWS account within five minutes after volume initialization completes. For more information, see Use an Amazon EBS Provisioned Rate for Volume Initialization.

The event is **not** sent for volumes created using Fast Snapshot Restore.

> **Important**
>
> `completionTime` reflects when we generated the event. Because the event is generated within 5 minutes after initialization completes, the `completionTime` can be up to five minutes after the initialization completed.

The following is an example event.

```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
   "detail-type": "EBS Volume Notification",
   "source": "aws.ec2",
   "account": "012345678901",
   "time": "yyyy-mm-ddThh:mm:ssZ",
   "region": "us-east-1",
   "resources": [
      "arn:aws:ec2:us-east-1:012345678901:volume/vol-01234567890abcdef"
   ],
   "detail": {
      "event": " initializeVolume",
      "result": "succeeded",
      "completionTime": "yyyy-mm-ddThh:mm:ssZ",
      "request-id": "01234567-0123-0123-0123-0123456789ab"
   }
}
```

EBS volume modification events

Amazon EBS sends `modifyVolume` events to EventBridge when a volume is modified. However, it is not saved, logged, or archived.


```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
   "detail-type": "EBS Volume Notification",
   "source": "aws.ec2",
   "account": "012345678901",
   "time": "yyyy-mm-ddThh:mm:ssZ",
   "region": "us-east-1",
   "resources": [
      "arn:aws:ec2:us-east-1:012345678901:volume/vol-03a55cf56513fa1b6"
   ],
   "detail": {
      "result": "optimizing",
      "cause": "",
      "event": "modifyVolume",
      "request-id": "01234567-0123-0123-0123-0123456789ab"
   }
}
```


EBS snapshot events

Amazon EBS sends events to EventBridge when the following volume events occur.

Events

- Create snapshot (createSnapshot)
- Create snapshots (createSnapshots)
- Copy snapshot (copySnapshot)
- Share snapshot (shareSnapshot)

Create snapshot (createSnapshot)

The `createSnapshot` event is sent to your AWS account when an action to create a snapshot completes. However, it is not saved, logged, or archived. This event can have a result of either `succeeded` or `failed`.

Event data

The listing below is an example of a JSON object emitted by EBS for a successful `createSnapshot` event. In the `detail` section, the `source` field contains the ARN of the source volume. The `startTime` and `endTime` fields indicate when creation of the snapshot started and completed.


```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
     "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "createSnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": "arn:aws:ec2::us-west-2:volume/vol-01234567",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ"  }
}
```


Create snapshots (createSnapshots)

The `createSnapshots` event is sent to your AWS account when an action to create a multi-volume snapshot completes. This event can have a result of either `succeeded` or `failed`.

Event data

The listing below is an example of a JSON object emitted by EBS for a successful `createSnapshots` event. In the `detail` section, the `source` field contains the ARNs of the source volumes of the multi-volume snapshot set. The `startTime` and `endTime` fields indicate when creation of the snapshot started and completed.


```
{
```

```
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Multi-Volume Snapshots Completion Status",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-east-1:snapshot/snap-01234567",
    "arn:aws:ec2::us-east-1:snapshot/snap-012345678"
  ],
  "detail": {
    "event": "createSnapshots",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ",
    "snapshots": [
      {
        "snapshot_id": "arn:aws:ec2::us-east-1:snapshot/snap-01234567",
        "source": "arn:aws:ec2::us-east-1:volume/vol-01234567",
        "status": "completed"
      },
      {
        "snapshot_id": "arn:aws:ec2::us-east-1:snapshot/snap-012345678",
        "source": "arn:aws:ec2::us-east-1:volume/vol-012345678",
        "status": "completed"
      }
    ]
  }
}
```


The listing below is an example of a JSON object emitted by EBS after a failed `createSnapshots` event. The cause for the failure was one or more snapshots for the multi-volume snapshot set failed to complete. The values of `snapshot_id` are the ARNs of the failed snapshots. `startTime` and `endTime` represent when the create-snapshots action started and ended.


```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Multi-Volume Snapshots Completion Status",
  "source": "aws.ec2",
```

```
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-east-1:snapshot/snap-01234567",
    "arn:aws:ec2::us-east-1:snapshot/snap-012345678"
  ],
 "detail": {
    "event": "createSnapshots",
    "result": "failed",
    "cause": "Snapshot snap-01234567 is in status error",
   "request-id": "",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ",
    "snapshots": [
      {
        "snapshot_id": "arn:aws:ec2::us-east-1:snapshot/snap-01234567",
        "source": "arn:aws:ec2::us-east-1:volume/vol-01234567",
        "status": "error"
      },
      {
        "snapshot_id": "arn:aws:ec2::us-east-1:snapshot/snap-012345678",
        "source": "arn:aws:ec2::us-east-1:volume/vol-012345678",
        "status": "error"
      }
    ]
  }
}
```


Copy snapshot (copySnapshot)

The `copySnapshot` event is sent to your AWS account when an action to copy a snapshot completes. However, it is not saved, logged, or archived. This event can have a result of either `succeeded` or `failed`.

In the `detail` section, `source` is the ARN of the source snapshot, and `snapshot_id` is the ARN of the snapshot copy. `startTime` and `endTime` indicate when the copy operation started and ended. `incremental` indicates whether the snapshot copy is an incremental snapshot (`true`), or a full snapshot (`false`). `transferType` indicates whether the snapshot copy operation was a standard copy operation or a time-based copy operation. For more information, see [Time-based copies for Amazon EBS snapshots and EBS-backed AMIs](#).

If you are copying the snapshot across Regions, then the event is emitted in the destination Region.

Scenario 1: Standard snapshot copy operation completes

The following is an example of an event that is sent to your account when a standard snapshot copy operation completes successfully. Note that `transferType` is `standard`.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "copySnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": "arn:aws:ec2::eu-west-1:snapshot/snap-76543210",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ",
    "incremental": "true",
    "transferType": "standard"
  }
}
```

Scenario 2: Time-based snapshot copy operation completes within completion duration

The following is an example of an event that is sent to your account when a time-based snapshot copy operation completes within its completion duration. Note that `transferType` is `time-based` to indicate that it was a time-based snapshot copy operation. `completionDurationStartTime` indicates when the completion duration started.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
```

```
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "copySnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": "arn:aws:ec2::eu-west-1:snapshot/snap-76543210",
    "incremental": "true",
    "completionDurationStartTime":"2024-11-16T06:27:33.816Z",
    "transferType": "time-based"
  }
}
```

Scenario 3: Time-based snapshot copy operation completes but misses the requested completion duration

When a time-based snapshot copy operation completes, but fails to meet the requested completion duration, CloudWatch sends two events to your account. The following are examples of those events.

- The first event is sent to your account as soon as the completion duration is missed, even if the copy operation is still in progress. For this event, the `detail-type` is `EBS Copy Snapshot Missed Completion Duration`, and `missedCompletionDurationCause` provides the reason.

  ```
  {
   "version":"0",
   "id":"fd90eb95-0938-e02c-cf55-b81363b8ac12",
   "detail-type":"EBS Copy Snapshot Missed Completion Duration",
   "source":"aws.ec2",
   "account":"123456789012",
   "time":"2024-11-19T18:17:08Z",
  ```

```
  "region":"us-east-1",
  "resources": ["arn:aws:ec2:us-east-1:123456789012:snapshot/snap-01234567890abcedf"],
  "detail":{
    "event":"copySnapshot",
    "missedCompletionDurationCause":"Snapshot copy was not able to meet the specified
  completion duration because your snapshot copy operation throughput quota was
  exceeded.",
    "snapshot_id":"arn:aws:ec2:us-
east-1:123456789012:snapshot/snap-01234567890abcedf",
    "source":"arn:aws:ec2:us-east-1:123456789012:snapshot/snap-00987654321fedcba",
    "startTime": "Sun Nov 24 22:32:55 UTC 2024",
    "transferType": "time-based"
    }
}
```

- The second event is sent to your account only once the snapshot is completed. The event includes `missedCompletionDurationCause`, which provides the reason.

```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "copySnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": "arn:aws:ec2::eu-west-1:snapshot/snap-76543210",
    "incremental": "true",
    "completionDurationStartTime":"2024-11-16T06:27:33.816Z",
    "missedCompletionDurationCause":"Snapshot copy was not able to meet the specified
  completion duration because your snapshot copy operation throughput quota was
  exceeded.",
```

```
    "transferType": "time-based"
  }
}
```

Scenario 4: Snapshot copy operation fails

The following is an example of an event that is sent to your account when a snapshot copy operation fails. Note that `result` is `failed` to indicate that the operation failed.


```
{
  "version": "0",
  "id": "01234567-0123-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "123456789012",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "copySnapshot",
    "result": "failed",
    "cause": "Source snapshot ID is not valid",
    "request-id": "",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": "arn:aws:ec2::eu-west-1:snapshot/snap-76543210",
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ"
  }
}
```


Share snapshot (shareSnapshot)

The `shareSnapshot` event is sent to your AWS account when another account shares a snapshot with it. However, it is not saved, logged, or archived. The result is always `succeeded`.

Event data

The following is an example of a JSON object emitted by EBS after a completed `shareSnapshot` event. In the `detail` section, the value of `source` is the AWS account number of the user that shared the snapshot with you. `startTime` and `endTime` represent when the share-snapshot action started and ended. The `shareSnapshot` event is emitted only when a private snapshot is shared with another user. Sharing a public snapshot does not trigger the event.

```
{
  "version": "0",
  "id": "01234567-01234-0123-0123-012345678901",
  "detail-type": "EBS Snapshot Notification",
  "source": "aws.ec2",
  "account": "012345678901",
  "time": "yyyy-mm-ddThh:mm:ssZ",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ec2::us-west-2:snapshot/snap-01234567"
  ],
  "detail": {
    "event": "shareSnapshot",
    "result": "succeeded",
    "cause": "",
    "request-id": "",
    "snapshot_id": "arn:aws:ec2::us-west-2:snapshot/snap-01234567",
    "source": 012345678901,
    "startTime": "yyyy-mm-ddThh:mm:ssZ",
    "endTime": "yyyy-mm-ddThh:mm:ssZ"
  }
}
```

EBS Snapshots Archive events

Amazon EBS emits events related to snapshot archiving actions. For more information, see [Monitor Amazon EBS snapshot archiving using CloudWatch Events](#).

EBS fast snapshot restore events

Amazon EBS sends events to EventBridge when the state of fast snapshot restore for a snapshot changes. Events are emitted on a best effort basis.

The following is example data for this event.

```
{
   "version": "0",
   "id": "01234567-0123-0123-0123-012345678901",
```

```
    "detail-type": "EBS Fast Snapshot Restore State-change Notification",
    "source": "aws.ec2",
    "account": "123456789012",
    "time": "yyyy-mm-ddThh:mm:ssZ",
    "region": "us-east-1",
    "resources": [
       "arn:aws:ec2:us-east-1::snapshot/snap-03a55cf56513fa1b6"
    ],
    "detail": {
       "snapshot-id": "snap-1234567890abcdef0",
       "state": "optimizing",
       "zone": "us-east-1a",
       "message": "Client.UserInitiated - Lifecycle state transition",
    }
}
```


The possible values for `state` are `enabling`, `optimizing`, `enabled`, `disabling`, and `disabled`.

The possible values for `message` are as follows:

`Client.InvalidSnapshot.InvalidState - The requested snapshot transitioned to an invalid state (Error)`

A request to enable fast snapshot restore failed and the state transitioned to `disabling` or `disabled`. Fast snapshot restore cannot be enabled for this snapshot.

`Client.UserInitiated`

The state successfully transitioned to `enabling` or `disabling`.

`Client.UserInitiated - Lifecycle state transition`

The state successfully transitioned to `optimizing`, `enabled`, or `disabled`.

`Server.InsufficientCapacity - There was insufficient capacity available to satisfy the request`

A request to enable fast snapshot restore failed due to insufficient capacity, and the state transitioned to `disabling` or `disabled`. Wait and then try again.

`Server.InternalError - An internal error caused the operation to fail`

A request to enable fast snapshot restore failed due to an internal error, and the state transitioned to `disabling` or `disabled`. Wait and then try again.

`Client.InvalidSnapshot.InvalidState - The requested snapshot was deleted or access permissions were revoked`

The fast snapshot restore state for the snapshot has transitioned to `disabling` or `disabled` because the snapshot was deleted or unshared by the snapshot owner. Fast snapshot restore cannot be enabled for a snapshot that has been deleted or is no longer shared with you.

Using AWS Lambda to handle EventBridge events

You can use Amazon EBS and Amazon EventBridge to automate your data-backup workflow. This requires you to create an IAM policy, a AWS Lambda function to handle the event, and an EventBridge rule that matches incoming events and routes them to the Lambda function.

The following procedure uses the `createSnapshot` event to automatically copy a completed snapshot to another Region for disaster recovery.

To copy a completed snapshot to another Region

1. Create an IAM policy, such as the one shown in the following example, to provide permissions to use the `CopySnapshot` action and write to the EventBridge log. Assign the policy to the user that will handle the EventBridge event.

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Action": [
           "logs:CreateLogGroup",
           "logs:CreateLogStream",
           "logs:PutLogEvents"
         ],
         "Resource": "arn:aws:logs:*:*:*"
       },
       {
         "Effect": "Allow",
         "Action": [
           "ec2:CopySnapshot"
   ```

```
        ],
        "Resource": "*"
      }
    ]
  }
```

2. Define a function in Lambda that will be available from the EventBridge console. The sample Lambda function below, written in Node.js, is invoked by EventBridge when a matching `createSnapshot` event is emitted by Amazon EBS (signifying that a snapshot was completed). When invoked, the function copies the snapshot from `us-east-2` to `us-east-1`.

```
// Sample Lambda function to copy an EBS snapshot to a different Region

var AWS = require('aws-sdk');
var ec2 = new AWS.EC2();

// define variables
var destinationRegion = 'us-east-1';
var sourceRegion = 'us-east-2';
console.log ('Loading function');

//main function
exports.handler = (event, context, callback) => {

    // Get the EBS snapshot ID from the event details
    var snapshotArn = event.detail.snapshot_id.split('/');
    const snapshotId = snapshotArn[1];
    const description = `Snapshot copy from ${snapshotId} in ${sourceRegion}.`;
    console.log ("snapshotId:", snapshotId);

    // Load EC2 class and update the configuration to use destination Region to
 initiate the snapshot.
    AWS.config.update({region: destinationRegion});
    var ec2 = new AWS.EC2();

    // Prepare variables for ec2.modifySnapshotAttribute call
    const copySnapshotParams = {
        Description: description,
        DestinationRegion: destinationRegion,
        SourceRegion: sourceRegion,
        SourceSnapshotId: snapshotId
    };
```

```
    // Execute the copy snapshot and log any errors
    ec2.copySnapshot(copySnapshotParams, (err, data) => {
        if (err) {
            const errorMessage = `Error copying snapshot ${snapshotId} to Region
 ${destinationRegion}.`;
            console.log(errorMessage);
            console.log(err);
            callback(errorMessage);
        } else {
            const successMessage = `Successfully started copy of snapshot
 ${snapshotId} to Region ${destinationRegion}.`;
            console.log(successMessage);
            console.log(data);
            callback(null, successMessage);
        }
    });
};
```

To ensure that your Lambda function is available from the EventBridge console, create it in the Region where the EventBridge event will occur. For more information, see the AWS Lambda Developer Guide.

3. Open the Amazon EventBridge console at https://console.aws.amazon.com/events/.
4. In the navigation pane, choose **Rules**, and then choose **Create rule**.
5. For **Step 1: Define rule detail**, do the following:
    a. Enter values for **Name** and **Description**.
    b. For **Event bus**, keep **default**.
    c. Ensure that **Enable the rule on the selected event bus** is toggled on.
    d. For **Event type**, select **Rule with an event pattern**.
    e. Choose **Next**.
6. For **Step 2: Build event pattern**, do the following:
    a. For **Event source**, select **AWS events or EventBridge partner events**.
    b. In the **Event pattern** section, for **Event source**, ensure that **AWS service** is selected, and for **AWS service**, select **EC2**.
    c. For **Event type**, select **EBS Snapshot Notification**, select **Specific event(s)**, and then choose **createSnapshot**.
    d. Select **Specific result(s)** and then choose **succeeded**.

   e. Choose **Next**.

7. For **Step 3: Select targets**, do the following:

   a. For **Target types**, choose **AWS service**.

   b. For **Select target**, choose **Lambda function**, and for **Function** select the function that you created earlier.

   c. Choose **Next**

8. For **Step 4: Configure tags**, specify tags for the rule if needed, and then choose **Next**.

9. For **Step 5: Review and create**, review the rule and then choose **Create rule**.

Your rule should now appear on the **Rules** tab. In the example shown, the event that you configured should be emitted by EBS the next time you copy a snapshot.

Amazon EBS detailed performance statistics

Amazon EBS NVMe block devices vend real-time, high-resolution I/O performance statistics for Amazon EBS volumes attached to Nitro-based Amazon EC2 instances. These statistics are presented as aggregated counters that are retained for the duration of the volume's attachment to the instance. The statistics provide details about the cumulative number of operations, bytes sent and received, and time spent on read and write I/O operations. Additionally, the statistics include histograms for read and write I/O operations, and the total time your application has exceeded the EBS volume or attached instance's provisioned IOPS or throughput limits.

You can collect these statistics at a granularity of up to 1 second intervals. If requests are made more frequently than 1 second intervals, the NVMe driver might queue the requests, along with other admin commands, to be processed at a later time.

Considerations

- The statistics are supported for all Amazon EBS volume types.
- The statistics are supported only for volumes attached to [instances built on the AWS Nitro System](#).
- The statistics are available for Multi-Attach enabled volumes. When viewing statistics for a Multi-Attach enabled volume, the statistics are specific to that instance attachment, and reflect only that instance's usage.
- The statistics are available at no additional cost.

Statistics

The Amazon EBS NVMe block device vends the following statistics:

Statistic name	Full name	Type	Description
`total_read_ops`	Total read operations	Counter	The total number of completed read operations.
`total_write_ops`	Total write operations	Counter	The total number of completed write operations.
`total_read_bytes`	Total read bytes	Counter	The total number of read bytes transferred.
`total_write_bytes`	Total write bytes	Counter	The total number of write bytes transferred.
`total_read_time`	Total read time	Counter	The total time spent, in microseconds, by all completed read operations.
`total_write_time`	Total write time	Counter	The total time spent, in microseconds, by all completed write operations.
`ebs_volume_performance_exceeded_iops`	Total time demand exceeded volume provisioned IOPS	Counter	The total time, in microseconds, that IOPS demand exceeded the volume's provisioned IOPS performance.
`ebs_volume_performance_exceeded_tp`	Total time demand exceeded volume provisioned throughput	Counter	The total time, in microseconds, that throughput demand exceeded the volume's provisioned throughput performance.
`ec2_instance_ebs_performan`	Total time demand exceeded EC2	Counter	The total time, in microseconds, that the EBS volume exceeded the attached Amazon EC2 instance's maximum IOPS performance.

Statistic name	Full name	Type	Description
e_exceeded_iops	instance's IOPS performance		
ec2_instance_ebs_performance_exceeded_tp	Total time demand exceeded EC2 instance's throughput performance	Counter	The total time, in microseconds, that the EBS volume exceeded the attached Amazon EC2 instance's maximum throughput performance.
volume_queue_length	Volume queue length	Point in time	The number of read and write operations waiting to be completed.
read_io_latency_histogram	Read I/O histogram	Histogram *	The number of read operations completed within each latency bin, in microseconds.
write_io_latency_histogram	Write I/O histogram	Histogram *	The number of write operations completed within each latency bin, in microseconds.

Note

* Histogram statistics represent only I/O operations that have completed successfully. Stalled or impaired I/O operations are not included, but will be evident in the `volume_queue_length` statistics, which is presented as a point-in-time statistic.

Accessing the statistics

The statistics must be accessed directly from the instance to which the Amazon EBS volume is attached. You can access the statistics using one of the following methods.

Linux instances

Amazon CloudWatch

You can configure the Amazon CloudWatch agent to collect the statistics from your instance and make them available as custom metrics in CloudWatch. You can then use the metrics in CloudWatch to analyze I/O patterns, track performance trends, create custom dashboards, and set up automated alarms based on performance thresholds.

For more information about configuring the CloudWatch agent, see the following:

- Create the CloudWatch agent configuration file
- Collect Amazon EBS NVMe driver metrics

With the Amazon CloudWatch Observability EKS add-on version `4.1.0` and later, the statistics are automatically collected when the Amazon EBS CSI driver metrics are enabled. For more information, see Amazon EBS NVMe driver metrics.

ebsnvme script

The `ebsnvme` script can be found in the amazon-ec2-utils Github repo.

To access the statistics

1. Connect to the instance to which the volume is attached.
2. Download the `ebsnvme` script from the `amazon-ec2-utils` Github repo.

```
wget https://raw.githubusercontent.com/amazonlinux/amazon-ec2-utils/refs/heads/main/ebsnvme
```

3. Modify the permissions for the script to make it executable.

```
sudo chmod +x ./ebsnvme
```

4. Run the `ebsnvme` script and specify the device name for the volume.

```
sudo ./ebsnvme stats /dev/nvme0n1
```

nvme-cli tool

To access the statistics

1. Connect to the instance to which the volume is attached.
2. Amazon Linux AMIs released after November 12, 2024 include the latest version of the `nvme-cli` tool. If you are using an older Amazon Linux AMI, update the `nvme-cli` tool.

```
sudo yum install nvme-cli
```

3. Run the following command and specify the device name for the volume.

```
nvme amzn stats /dev/nvme0n1
```

Prometheus

You can monitor the statistics with Prometheus, an open-source monitoring application, and Amazon Managed Service for Prometheus. This makes it easier to monitor Amazon EBS volumes across container and Kubernetes environments at scale. With Amazon EBS CSI driver version v1.37.0 and later, the detailed performance statistics are exposed as a Prometheus-compatible `/metrics` endpoint for exporting into Prometheus.

For more information, see Ingest metrics to your Amazon Managed Service for Prometheus workspace in the *Amazon Managed Service for Prometheus User Guide*.

Windows instances

nvme_amzn.exe tool

To access the statistics

1. Connect to the instance to which the volume is attached.
2. Make sure that you're using AWSNVMe driver version `1.7.0` or later. For more information about updating the AWSNVMe driver, see AWS NVMe drivers.
3. Get the disk number for the EBS volume. For more information, see Map Amazon EBS volumes to NVMe device names
4. Run the following command as Administrator and specify the disk number for the volume.

```
.\nvme_amzn.exe stats disk_number
```

Amazon GuardDuty for Amazon EBS

Amazon GuardDuty is a threat detection service that helps protect your accounts, containers, workloads, and the data within your AWS environment. Using machine learning (ML) models, and anomaly and threat detection capabilities, GuardDuty continuously monitors different log sources and runtime activity to identify and prioritize potential security risks and malicious activities in your environment.

The Malware Protection feature within GuardDuty scans the Amazon EBS volumes associated with your Amazon EC2 instances and container workloads to detect potential threats. GuardDuty offers two ways to do this:

- **Enable Malware Protection** — When GuardDuty generates a finding that is indicative of potential presence of malware in an Amazon EC2 instance or a container workload, it will automatically initiate a malware scan on the potentially compromised resource.
- **Use on-demand malware scan without enabling Malware Protection** — Provide the Amazon Resource Name (ARN) of your Amazon EC2 instance to initiate an on-demand scan.

For more information, see the Amazon GuardDuty User Guide.

Understand codes for Amazon EBS in billing and usage reports

When you use Amazon EBS, we include related codes in your AWS billing and usage reports. Reviewing these codes helps you understand your costs and usage patterns for Amazon EBS. Tracking and managing your expenses is essential for optimizing your costs.

The following tables describe the codes for Amazon EBS that appear in your billing and usage reports. For a list of the Region codes used in the billing and usage reports, see AWS Region billing codes.

Billing codes for:

- Snapshots
- Volume storage

- Provisioned performance
- EBS direct APIs
- Fast snapshot restore (FSR)

Related resources

- Amazon EBS pricing
- the section called “Pricing”

Snapshots

Code	Description	Units	Granularity
region-EBS:SnapshotUsage	Standard tier snapshots (incremental copies).	GB-Month	Hourly
region-EBS:SnapshotArchiveStorage	Archive tier snapshots (full copies).	GB-Month	Hourly
region-EBS:SnapshotArchiveEarlyDelete	Deleting archived snapshots before the minimum retention period ends.	GB	Per deletion
region-EBS:SnapshotArchiveRetrieval	Retrieving archived snapshots.	GB	Per GB

Volume storage

Code	Description	Units	Granularity
region-EBS:VolumeUsage.gp3	Storage for General Purpose SSD (gp3) volumes.	GB-Month	Hourly
region-EBS:VolumeUsage.gp2	Storage for General Purpose SSD (g2) volumes.	GB-Month	Hourly
region-EBS:VolumeUsage.io2	Storage for Provisioned IOPS SSD (io2) volumes.	GB-Month	Hourly
region-EBS:VolumeUsage.piops	Storage for Provisioned IOPS SSD (io1) volumes.	GB-Month	Hourly
region-EBS:VolumeUsage.st1	Storage for Throughput Optimized HDD volumes.	GB-Month	Hourly
region-EBS:VolumeUsage.sc1	Storage for Cold HDD volumes.	GB-Month	Hourly
region-EBS:VolumeUsage	Storage for Magnetic volumes.	GB-Month	Hourly

Provisioned performance

Code	Description	Units	Granularity
region-EBS:VolumeP-Throughput.gp3	Provisioned throughput for gp3 volumes over 125 MB/s.	MB/s-Month	Hourly
region-EBS:VolumeP-IOPS.gp3	Provioned IOPS for gp3 volumes over 3,000.	IOPS-Month	Hourly
region-EBS:VolumeP-IOPS.io2	Provisioned IOPS for io2 volumes up to 32,000.	IOPS-Month	Hourly
region-EBS:VolumeP-IOPS.io2.tier2	Provisioned IOPS for io2 volumes from 32,001 to 64,000.	IOPS-Month	Hourly
region-EBS:VolumeP-IOPS.io2.tier3	Provisioned IOPS for io2 volumes greater than 64,000.	IOPS-Month	Hourly
region-EBS:VolumeP-IOPS.piops	Provisioned IOPS for io1 volumes.	IOPS-Month	Hourly
region-EBS:VolumeIOUsage	Legacy IOPS.	IOPS-Month	Hourly

EBS direct APIs

Code	Description	Units	Granularity
region-EBS:directAPI.snapshot.List	Calls to the `ListChangedBlocks` and `ListSnapshotBlocks` API actions.	Per 1,000 requests	Per request
region-EBS:directAPI.snapshot.Get	Blocks returned by the `GetSnapshotBlock` API action.	Per 1,000 blocks returned	Per request
region-EBS:directAPI.snapshot.Put	Blocks written by the `PutSnapshotBlock` API action.	Per 1,000 blocks written	Per request

Fast snapshot restore (FSR)

Code	Description	Units	Granularity
region-EBS:FastSnapshotRestore	Data Services Unit-Hours (DSU-Hours) for snapshots enabled for fast snapshot restore (per AZ).	DSU-Hours	Per minute (one-hour minimum)

Create an inventory of your EBS volumes

Amazon EBS provides scalable, high-performance block storage resources that can be used with EC2 instances. You can attach an EBS volume to an EC2 instance. You can also detach an EBS volume from an EC2 instance.

The following are the key characteristics of EBS storage. A system administrator can get information about these characteristics for your EBS resources and use it to configure functionally equivalent storage for servers that you run on premises or servers from another Cloud Provider. A system administrator can also retrieve the data stored in your EBS resources and then store the data on premises or in storage from another Cloud Provider.

Characteristic	Description
Volumes	An EBS volume is a durable, block-level storage device that you can attach to an instance, format, and mount.
I/O performance	Amazon EBS provides multiple volume types, which differ in terms of performance character istics and price.
Contents of your EBS data volumes	The only way to directly access the data on a volume is from the instance.
Snapshots	An EBS snapshot is a point-in-time copy of the data on an EBS volume. Snapshots are stored in Amazon S3, in buckets that you can't access directly.
Contents of your EBS snapshots	You can read data from your snapshots, or use them to create volumes.

Volumes

When you create an EBS volume, you select an Availability Zone for the volume. You must attach an EBS volume to an EC2 instance in the same Availability Zone. You can create volumes when you launch an EC2 instance, or create them first and then attach them to an instance. If a volume is attached to an instance, its status is `in-use`. If a volume is not attached to an instance, its status is `available`.

Why these matter

EBS root volumes contain the operating system for your EC2 instances. EBS data volumes can contain business critical data. You can list your EBS volumes across all Regions and whether they are attached to EC2 instances.

To get a summary of your EBS volumes across all Regions

You can use Amazon EC2 Global View to list your EBS volumes across all Regions.

1. Open the Amazon EC2 Global View console at https://console.aws.amazon.com/ec2globalview/home.
2. On the **Region explorer** tab, under **Summary**, check the resource count for **Volumes**, which includes the number of volumes and the number of Regions. Click the underlined text to see how the volume count is spread across Regions.
3. On the **Global search** tab, select the client filter **Resource type = Volume**. You can filter the results further by specifying a Region or a tag.

To describe the EBS volumes that are in use

Use the describe-volumes command. You must run this command in each Region where you have EBS volumes. The `--filters` parameter scopes the results to volumes that are `in-use`. The `--query` parameter displays only the specified fields in the output. You can include additional fields as needed.

```
aws ec2 describe-volumes \
    --filters Name=status,Values=in-use \
    --query "Volumes[].[VolumeId,Size,Attachments[0].InstanceId]" \
    --output table
```

The following is example output. The columns are volume ID, volume size, ID of the attached instance, and the device name.

```
----------------------------------------------------------------------
|                          DescribeVolumes                           |
+-------------------------+-----+-----------------------+------------+
|  vol-0992ee65bec96de19 |  8  |  i-00a7d9ec76a46a49f  |  /dev/xvda |
|  vol-04d631d22047db1cb |  30 |  i-0b1bf24fd4f297ab9  |  /dev/sda1 |
   ...
   ...
|  vol-0b811be5cbd1425cc | 100 |  i-0a8d998154b320257  |  /dev/xvde |
```

```
+------------------------+-----+-----------------------+------------+
```

To describe the EBS volumes that are not in use

In the previous command, modify the filter to find volumes with a status of `available` as follows.

```
--filters Name=status,Values=available
```

If a volume is unused, you should determine whether it has data that you need. For example, you can delete volumes that were created for testing purposes or to troubleshoot an issue. A volume might also be available if the block device that attached the volume to the instance was configured to persist after instance termination. If this is the case, verify whether the data on the volume is still needed.

I/O performance

The volume type that you specify for an EBS volume determines its performance. Depending on the volume type, you can specify a combination of size, IOPS, and throughput for your volumes. For information about the maximum provisioned IOPS and throughput for each volume type, see Amazon EBS volume types.

Why this matters

After you determine the performance of your EBS volumes, you can decide the minimum specifications required for functionally equivalent storage.

To get information about the performance of your volumes

Use the describe-volumes command to describe the performance for your volumes that are in use. The `--query` parameter displays only the specified fields in the output. You can include additional fields as needed. For information about EBS-optimized performance, refer to Amazon EC2 instance type specifications in the *Amazon EC2 Instance Types Guide*.

```
aws ec2 describe-volumes \
    --filters Name=status,Values=in-use \
    --query "Volumes[].[VolumeId,VolumeType,Iops,Throughput]" \
    --output table
```

The following is example output. The columns are volume ID, volume type, IOPS, and throughput (MiB/s).

```
-------------------------------------------------
|                DescribeVolumes                |
+-----------------------+------+-------+-------+
|  vol-0992ee65bec96de19 |  gp3 |  3000 |  125  |
|  vol-04d631d22047db1cb |  gp2 |  100  |  None |
   ...
   ...
|  vol-0b811be5cbd1425cc |  gp3 |  3000 |  125  |
+-----------------------+------+-------+-------+
```

Contents of your EBS data volumes

Amazon EBS does not have access to the data that you store on your EBS volumes. You are responsible for backing up your data volumes. For example, you can create point-in-time snapshots, known as EBS snapshots.

You can also connect to your EC2 instances and transfer files from your EBS volumes to your own computer. There are many tools available to help you do this. The following are examples.

Why this matters

If you have business critical data on your EBS volumes, you can copy it to functionally equivalent storage.

To transfer files from a volume attached to a Linux instance

Connect to your Linux instance and use SCP. For more information, see Transfer files using SCP.

To transfer files from a volume attached to a Windows instance

Connect to your Windows instance and use RDP. For more information, see Transfer files using RDP.

Snapshots

You can back up your EBS data volumes by creating EBS snapshots. You can create EBS snapshots using Amazon EBS, Amazon Data Lifecycle Manager, or AWS Backup.

Why this matters

If you have business critical data on your EBS volumes, most likely you're creating backups in the form of EBS snapshots. You can find the most recent snapshots of your volumes and verify that they reflect the current data on the volumes.

To describe the snapshots for a specific volume

Use the describe-snapshots command. The `--filters` parameter scopes the results to snapshots for the specified volume. The `--query` parameter displays only the specified fields in the output. You can include additional fields as needed.

```
aws ec2 describe-snapshots \
    --filters Name=volume-id,Values=vol-00622ef8c2ac8b762 \
    --query Snapshots[*].[SnapshotId,CompletionTime,Description] --output table
```

The following is example output. The columns are snapshot ID, completion date, and whether the volume is encrypted.

```
-------------------------------------------------------------------------
|                          DescribeSnapshots                            |
+-------------------------+-----------------------------------+--------+
|  snap-0ad439c50efabb47c |  2024-07-12T03:36:27.952000+00:00  |  False |
+-------------------------+-----------------------------------+--------+
```

Contents of your EBS snapshots

You can't download or export the contents of an EBS snapshot to another format. However, you can access the data in your snapshots by using the EBS direct APIs, or by creating an EBS volume from the snapshot and transferring the files.

Why this matters

If you have business critical data on your EBS volumes, most likely you're creating backups in the form of EBS snapshots. You can retrieve this data and copy it to functionally equivalent storage.

Option 1: To read data from your snapshots

Use the EBS direct APIs.

Option 2: To create a volume from a snapshot and transfer the data

1. Use the create-volume command to create a volume from the snapshot.

```
aws ec2 create-volume \
   --volume-type gp3 \
   --snapshot-id snap-0ad439c50efabb47c \
```

```
    --iops 6000 \
    --availability-zone us-east-2b
```

2. Use the attach-volume command to attach the volume to an instance.

```
aws ec2 attach-volume \
    --device /dev/sdf \
    --instance-id i-0b1bf24fd4f297ab9 \
    --volume-id vol-1234567890abcdef0
```

3. Connect to your instance and format and mount the volume.
4. Transfer the files on the volume to your own computer.

Quotas for Amazon EBS

Your AWS account has default quotas, formerly referred to as limits, for each AWS service. Unless otherwise noted, each quota is Region-specific. You can request increases for some quotas, and other quotas cannot be increased.

To view the quotas for Amazon EBS, open the Service Quotas console. In the navigation pane, choose **AWS services** and select **Amazon Elastic Block Store (Amazon EBS)**. To request a quota increase, see Requesting a Quota Increase in the *Service Quotas User Guide*.

Your AWS account has the following quotas related to Amazon EBS.

Name	**Default**	**Adjustable**	**Description**
Archived snapshots per volume	Each supported Region: 25	Yes	The maximum number of archived snapshots per volume.
CompleteSnapshot requests per account	Each supported Region: 10 per second	No	The maximum number of CompleteSnapshot requests allowed per account.
Concurrent snapshot copies per destination Region	Each supported Region: 20	No	The maximum number of concurrent snapshot copies to a single destination Region.
Concurrent snapshots per Cold HDD (sc1) volume	Each supported Region: 1	No	The maximum number of concurrent snapshots per Cold HDD (sc1) volume in this Region.
Concurrent snapshots per General Purpose SSD (gp2) volume	Each supported Region: 5	No	The maximum number of concurrent snapshots per General Purpose

Name	Default	Adjustable	Description
			SSD (gp2) volume in this Region.
Concurrent snapshots per General Purpose SSD (gp3) volume	Each supported Region: 5	No	The maximum number of concurrent snapshots per General Purpose SSD (gp3) volume in this Region.
Concurrent snapshots per Magnetic (standard) volume	Each supported Region: 5	No	The maximum number of concurrent snapshots per Magnetic (standard) volume in this Region.
Concurrent snapshots per Provisioned IOPS SSD (io1) volume	Each supported Region: 5	No	The maximum number of concurrent snapshots per Provisioned IOPS SSD (io1) volume in this Region.
Concurrent snapshots per Provisioned IOPS SSD (io2) volume	Each supported Region: 5	No	The maximum number of concurrent snapshots per Provisioned IOPS SSD (io2) volume in this Region.
Concurrent snapshots per Throughput Optimized HDD (st1) volume	Each supported Region: 1	No	The maximum number of concurrent snapshots per Throughput Optimized HDD (st1) volume in this Region.

Name	Default	Adjustable	Description
Fast snapshot restore	us-east-1: 5 us-east-2: 5 us-west-1: 5 us-west-2: 5 af-south-1: 5 ap-east-1: 5 ap-northeast-1: 5 ap-northeast-2: 5 ap-northeast-3: 5 ap-south-1: 5 ap-southeast-1: 5 ap-southeast-2: 5 ap-southeast-3: 5 ca-central-1: 5 eu-central-1: 5 eu-north-1: 5 eu-south-1: 5 eu-west-1: 5 eu-west-2: 5 eu-west-3: 5	Yes	The maximum number of snapshots that can be enabled for fast snapshot restore in this Region.

Name	Default	Adjustable	Description
	me-south-1: 5 sa-east-1: 5 Each of the other supported Regions: 5		
GetSnapshotBlock requests per account	us-east-1: 5,000 per second us-east-2: 5,000 per second us-west-2: 5,000 per second ap-southeast-1: 5,000 per second eu-west-1: 5,000 per second Each of the other supported Regions: 1,000 per second	Yes	The maximum number of GetSnapshotBlock requests allowed per account.
GetSnapshotBlock requests per snapshot	Each supported Region: 1,000 per second	No	The maximum number of GetSnapshotBlock requests allowed per snapshot.

Name	Default	Adjustable	Description
IOPS for Provisioned IOPS SSD (io1) volumes	Each supported Region: 300,000	Yes	The maximum aggregated number of IOPS that can be provisioned across Provisioned IOPS SDD (io1) volumes in this Region.
IOPS for Provisioned IOPS SSD (io2) volumes	Each supported Region: 100,000	Yes	The maximum aggregated number of IOPS that can be provisioned across Provisioned IOPS SDD (io2) volumes in this Region.
IOPS modifications for Provisioned IOPS SSD (io1) volumes	Each supported Region: 500,000	Yes	The maximum aggregated number of IOPS that can be requested in volume modifications across Provisioned IOPS SSD (io1) volumes in this Region.
IOPS modifications for Provisioned IOPS SSD (io2) volumes	Each supported Region: 100,000	Yes	The maximum current (from) and requested (to) IOPS for volume modification requests across Provisioned IOPS SSD (io2) volumes in this Region.
In-progress snapshot archives per account	Each supported Region: 25	Yes	The maximum number of in-progress snapshot archives per account.

Name	Default	Adjustable	Description
In-progress snapshot restores from archive per account	Each supported Region: 5	Yes	The maximum number of in-progress snapshot restores from archive per account.
ListChangedBlocks requests per account	Each supported Region: 50 per second	No	The maximum number of ListChangedBlocks requests allowed per account.
ListSnapshotBlocks requests per account	Each supported Region: 50 per second	No	The maximum number of ListSnapshotBlocks requests allowed per account.
Max concurrent copy volume operations per account	Each supported Region: 5	Yes	Maximum number of concurrent copy volume operations allowed per account in this region.
Provisioned Rate for Volume Initialization across concurrent volume creation requests per Region	Each supported Region: 5,000	Yes	The maximum aggregated Provisioned Rate for Volume Initialization, in MiB/s, that can be requested across concurrent volume creation requests in this Region.

Name	Default	Adjustable	Description
PutSnapshotBlock requests per account	us-east-1: 5,000 per second us-east-2: 5,000 per second us-west-2: 5,000 per second ap-southeast-1: 5,000 per second eu-west-1: 5,000 per second Each of the other supported Regions: 1,000 per second	Yes	The maximum number of PutSnapshotBlock requests allowed per account.
PutSnapshotBlock requests per snapshot	Each supported Region: 1,000 per second	No	The maximum number of PutSnapshotBlock requests allowed per snapshot.
Snapshots per Region	Each supported Region: 100,000	Yes	The maximum number of snapshots per Region.
StartSnapshot pending snapshots per account	Each supported Region: 100	No	The maximum number of pending snapshots per account that can be created using the StartSnapshot API.

Name	Default	Adjustable	Description
StartSnapshot requests per account	Each supported Region: 10 per second	No	The maximum number of StartSnapshot requests allowed per account.
Storage for Cold HDD (sc1) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Cold HDD (sc1) volumes in this Region.

Name	Default	Adjustable	Description
Storage for General Purpose SSD (gp2) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across General Purpose SSD (gp2) volumes in this Region.
Storage for General Purpose SSD (gp3) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across General Purpose SSD (gp3) volumes in this Region.

Name	Default	Adjustable	Description
Storage for Magnetic (standard) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Magnetic (standard) volumes in this Region.
Storage for Provisioned IOPS SSD (io1) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Provisioned IOPS SSD (io1) volumes in this Region.

Name	Default	Adjustable	Description
Storage for Provisioned IOPS SSD (io2) volumes, in TiB	Each supported Region: 20	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Provisioned IOPS SSD (io2) volumes in this Region.
Storage for Throughput Optimized HDD (st1) volumes, in TiB	af-south-1: 300 ap-east-1: 300 ap-northeast-3: 300 ap-southeast-3: 300 eu-south-1: 300 me-south-1: 300 Each of the other supported Regions: 50	Yes	The maximum aggregated amount of storage, in TiB, that can be provisioned across Throughput Optimized HDD (st1) volumes in this Region.
Storage modifications for Cold HDD (sc1) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across Cold HDD (sc1) volumes in this Region.

Name	Default	Adjustable	Description
Storage modifications for General Purpose SSD (gp2) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across General Purpose SSD (gp2) volumes in this Region.
Storage modifications for General Purpose SSD (gp3) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across General Purpose SSD (gp3) volumes in this Region.
Storage modifications for Magnetic (standard) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across Magnetic (standard) volumes in this Region.
Storage modifications for Provisioned IOPS SSD (io1) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across Provisioned IOPS SSD (io1) volumes in this Region.

Name	Default	Adjustable	Description
Storage modifications for Provisioned IOPS SSD (io2) volumes, in TiB	Each supported Region: 20	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across Provisioned IOPS SSD (io2) volumes in this Region.
Storage modifications for Throughput Optimized HDD (st1) volumes, in TiB	Each supported Region: 500	Yes	The maximum aggregated amount of storage, in TiB, that can be requested in volume modifications across Throughput Optimized HDD (st1) volumes in this Region.
Time-based snapshot copy throughput per destination Region	Each supported Region: 2,000	Yes	The maximum account-level throughput, in MiB/sec, for time-based snapshot copy operations per destination Region.

Considerations

- Your quotas can change over time. Amazon EBS constantly monitors your provisioned storage and IOPS usage within each Region and might automatically increase your quotas, on a per-Region basis, based on your usage. Even though Amazon EBS can automatically increase your quotas based on your usage, you can request a quota increase if needed. For example, if you plan to use more gp3 storage in US East (N. Virginia) than your current quota, you can request a quota increase for that volume type in that Region ahead of your planned usage.

- The quota for **Concurrent snapshot copies per destination Region** is not adjustable using Service Quotas. However, you can request an increase for this quota by contacting AWS Support.
- The **IOPS modifications and Storage modifications** quotas apply to the aggregated current value (for *size* or *IOPS*, depending on the quota) of volumes that can undergo modifications concurrently. You can make concurrent modification requests for volumes that have combined current value (for *size* or *IOPS*) up to the quota. For example, if your **IOPS modifications for Provisioned IOPS SSD (io1) volumes** quota is `50,000`, you can make concurrent IOPS modifications requests for any number of `io1` volumes as long as their combined current IOPS is equal to or less than `50,000`. If you have three `io1` volumes provisioned with `20,000` IOPS each, you can request IOPS modifications for two volumes concurrently (`20,000 * 2 < 50,000`). If you submit a concurrent IOPS modification request for the third volume, you exceed your quota and that request fails (`20,000 * 3 > 50,000`).
- Amazon EBS has the following non-adjustable limits for the number of EBS volumes per instance launch request.
  - `2500` — `us-east-1`, `us-west-2`, `eu-west-1`, and `ap-northeast-1`
  - `500` — all other Regions

  This limit applies to instance launch requests that you make, and to instance launch requests made by AWS services, such as Amazon EMR, on your behalf. If your instance launch request fails as a result of exceeding this limit, we recommend that you adjust the EBS volume configuration in the launch request to ensure the number of volumes is below the limit, or that you work with your technical account manager (TAM) to explore other options for launching your cluster without exceeding the limit.

Document history for the Amazon EBS User Guide

The following table describes the documentation releases for Amazon EBS.

Change	Description	Date
Elastic Volumes modification update	You can now initiate a new Elastic Volumes modification as soon as the previous one completes, up to four times within a 24-hour period.	January 15, 2026
Recycle Bin for EBS volumes	Recycle Bin enables you to protect Amazon EBS volumes against accidental deletions.	November 20, 2025
Copy an EBS volume	You can create an instant, point-in-time copy of an Amazon EBS volume within the same Availability Zone.	October 14, 2025
gp3 volume updates	You can now provision General Purpose SSD (gp3) volumes with sizes up to 64 TiB, IOPS up to 80,000, and throughput up to 2,000 MiB/s.	September 26, 2025
Latency guidance updates for io2 volumes	`io2` Block Express volumes are designed to deliver an average latency of under 500 microseconds for 16KiB I/O operations. `io2` Block Express volumes also deliver better outlier latency compared to General Purpose volumes, reducing the frequency of I/	September 26, 2025

	Os exceeding 800 microseconds by over 10 times.	
Latency Injection	Use the AWS FIS latency Injection action to simulate elevated I/O latency on your Amazon EBS volumes.	September 16, 2025
`io2` Region expansion	Provisioned IOPS SSD (`io2`) volumes are now available in all commercial and AWS GovCloud (US) Regions.	July 22, 2025
Local snapshots in Local Zones	You can now create local snapshots in Local Zones.	May 15, 2025
Amazon EBS Provisioned Rate for Volume Initialization	You can now specify a volume initialization rate when you create Amazon EBS volumes to ensure that they are fully initialized in a predictable amount of time.	May 6, 2025
Amazon Data Lifecycle Manager VPC endpoints	You can now establish a private connection between your VPC and Amazon Data Lifecycle Manager by creating an interface VPC endpoint.	February 28, 2025
Time-based AMI copies	You can now request a completion duration for EBS-backed AMI copy operations to ensure that AMI copies are completed within a specific timeframe.	February 25, 2025

Full snapshot size	You can now view the full size of an Amazon EBS snapshot using the Amazon EC2 console and AWS CLI.	February 11, 2025
Amazon Data Lifecycle Manager IPv6 support	Amazon Data Lifecycle Manager now provides dual-stack endpoints that support both IPv4 and IPv6 traffic.	February 7, 2025
Recycle Bin IPv6 support	Recycle Bin now provides dual-stack endpoints that support both IPv4 and IPv6 traffic.	December 19, 2024
Local snapshots in Dedicated Local Zones	You can now create local snapshots in Dedicated Local Zones.	December 16, 2024
AWSDataLifecycleManagerServiceRole AWS managed policy updated	The AWSDataLifecycleManagerServiceRole AWS managed policy has been updated to include permission for the `ec2:DescribeAvailabilityZones` action.	December 16, 2024
Declarative policies for block public access for EBS snapshots	You can now use declarative policies to apply account-level settings for block public access for snapshots across multiple Regions and accounts simultaneously. For more information, see Declarative policies in the *AWS Organizations User Guide*.	December 1, 2024

Time-based snapshot copies	You can now request a completion duration for snapshot copy operations to ensure that snapshot copies are completed within a specific timeframe.	November 26, 2024
Exclusion tags for Recycle Bin	You can now add exclusion tags to Region-level retention rules to exclude resources that have specific tags.	November 19, 2024
CloudFormation support for Recycle Bin	You can now create and manage Recycle Bin retention rules using AWS CloudFormation.	November 18, 2024
Amazon EBS detailed performance statistics	Amazon EBS NVMe block devices vend real-time, high-resolution I/O performance statistics for Amazon EBS volumes attached to Nitro-based Amazon EC2 instances.	November 12, 2024
New CloudWatch metrics for Amazon EBS volumes	You can now use the `VolumeAvgReadLatency` , `VolumeAvgWriteLatency` , `VolumeIOPSExceededCheck` , and `VolumeThroughputExceededCheck` Amazon CloudWatch metrics to monitor volume performance.	October 30, 2024

Enable Amazon Data Lifecycle Manager default policies across accounts	You can use CloudFormation StackSets to enable Amazon Data Lifecycle Manager default policies across an AWS organization or across specific AWS accounts.	April 26, 2024
AWSDataLifecycleManagerSSMFullAccess AWS managed policy	Updated the policy to support application-consistent snapshots for SAP HANA using the `AWSSystemsManagerSAP-CreateDLMSnapshotForSAPHANA` SSM document.	November 17, 2023
VolumeStalledIOCheck metric	You can use the `VolumeStalledIOCheck` metric to check whether a volume has passed or failed a stalled IO check in the last minute.	November 16, 2023
Amazon Data Lifecycle Manager default policies	You can now create Amazon Data Lifecycle Manager default policies for EBS snapshots and EBS-backed AMIs to backup all volumes and instances in a Region.	November 16, 2023
Amazon EBS snapshot lock	You can lock your Amazon EBS snapshots to protect them against accidental or malicious deletions, or to or store them in WORM format for a specific duration.	November 15, 2023

Block public access for snapshots	You can now use block public access for snapshots to prevent the public sharing of your snapshots.	November 9, 2023
Amazon Data Lifecycle Manager pre and post scripts	You can now use pre and post scripts in your Amazon Data Lifecycle Manager snapshot policies to automate the lifecycle of application-consis tent snapshots.	November 7, 2023
NVMe reservations	Multi-Attach enabled `io2` volumes support NVMe reservations, which is a set of industry-standard storage fencing protocols.	September 18, 2023
Fault testing on Amazon EBS	Use AWS FIS to temporari ly stop I/O between an EBS volume and the instances to which it is attached to test how your workloads handle I/ O interruptions.	January 27, 2023
Recycle Bin retention rule lock	You can lock retention rules to help protect them against accidental or malicious modifications and deletions.	November 23, 2022
Condition keys for Recycle Bin	You can use the `rbin:Requ est/ResourceType` and `rbin:Attribute/Res ourceType` condition keys to filter access on Recycle Bin requests.	June 14, 2022

io2 Block Express volumes	You can modify the size and provisioned IOPS of io2 Block Express volumes and you can enable them for fast snapshot restore.	May 31, 2022
Recycle Bin for AMIs	Recycle Bin enables you to restore accidentally deleted AMIs.	February 3, 2022
Recycle Bin for Amazon EBS snapshots	Recycle Bin for Amazon EBS snapshots is a snapshot recovery feature that enables you to restore accidentally deleted snapshots.	November 29, 2021
Amazon EBS Snapshots Archive	Amazon EBS Snapshots Archive is a new storage tier that you can use for low-cost, long-term storage of your rarely-accessed snapshots.	November 29, 2021
AMI deprecation support for Amazon Data Lifecycle Manager	Amazon Data Lifecycle Manager EBS-backed AMI policies can deprecate AMIs. The AWSDataLifecycleManagerServiceRoleForAMIManagement AWS managed policy has been updated to support this feature.	August 23, 2021
CloudWatch metrics for Amazon Data Lifecycle Manager	You can monitor your Amazon Data Lifecycle Manager policies using Amazon CloudWatch.	July 28, 2021

CloudTrail data events for EBS direct APIs	The ListSnapshotBlocks , ListChangedBlocks, GetSnapshotBlock, and PutSnapshotBlock APIs can be logged data events in CloudTrail.	July 27, 2021
`io2` Block Express volumes	`io2` Block Express volumes are now generally available.	July 19, 2021
Amazon EBS local snapshots on Outposts	You can now use Amazon EBS local snapshots on Outposts to store snapshots of volumes on an Outpost locally in Amazon S3 on the Outpost itself.	February 4, 2021
Multi-Attach support for io2 volumes	You can now enable Provision ed IOPS SSD (`io2`) volumes for Amazon EBS Multi-Attach.	December 18, 2020
Amazon Data Lifecycle Manager	Use Amazon Data Lifecycle Manager to automate the process of sharing snapshots and copying them across AWS accounts.	December 17, 2020
gp3 volumes	A new Amazon EBS General Purpose SSD volume type. You can specify provisioned IOPS and throughput when you create or modify the volume.	December 1, 2020
Throughput Optimized HDD and Cold HDD volume sizes	Throughput Optimized HDD (`st1`) and Cold HDD (`sc1`) volumes can range in size from 125 GiB to 16 TiB.	November 30, 2020

Amazon Data Lifecycle Manager	You can use Amazon Data Lifecycle Manager to automate the creation, retention, and deletion of EBS-backed AMIs.	November 9, 2020
Amazon Data Lifecycle Manager	Amazon Data Lifecycle Manager policies can be configured with up to four schedules.	September 17, 2020
Provisioned IOPS SSD (io2) volumes for Amazon EBS	Provisioned IOPS SSD (`io2`) volumes are designed to provide 99.999 percent volume durability with an AFR no higher than 0.001 percent.	August 24, 2020
Fast snapshot restore	You can enable fast snapshot restore for snapshots that are shared with you.	July 21, 2020
Amazon EBS Multi-Attach	You can now attach a single Provisioned IOPS SSD (io1) volume to up to 16 Nitro-based instances that are in the same Availability Zone.	February 14, 2020
Amazon EBS fast snapshot restores	You can enable fast snapshot restores on an EBS snapshot to ensure that EBS volumes created from the snapshot are fully-initialized at creation and instantly deliver all of their provisioned performance.	November 20, 2019

Amazon EBS multi-volume snapshots	You can take exact point-in-time, data coordinated, and crash-consistent snapshots across multiple EBS volumes attached to an EC2 instance.	May 29, 2019
Amazon EBS encryption by default	After you enable encryption by default in a Region, all new EBS volumes you create in the Region are encrypted using the default KMS key for EBS encryption.	May 23, 2019
Automate snapshot lifecycle	You can use Amazon Data Lifecycle Manager to automate creation and deletion of snapshots for your EBS volumes.	July 12, 2018
Perform modifications on attached EBS volumes	With most EBS volumes attached to most EC2 instances, you can modify volume size, type, and IOPS without detaching the volume or stopping the instance.	February 13, 2017
Copy encrypted Amazon EBS snapshots between AWS accounts	You can now copy encrypted EBS snapshots between AWS accounts.	June 21, 2016
Throughput Optimized HDD and Cold HDD volume types	You can now create Throughput Optimized HDD (st1) and Cold HDD (sc1) volumes.	April 19, 2016

General Purpose SSD volume type	General Purpose SSD volumes offer cost-effective storage that is ideal for a broad range of workloads. These volumes deliver single-digit millisecond latencies, the ability to burst to 3,000 IOPS for extended periods of time, and a base performance of 3 IOPS/GiB. General Purpose SSD volumes can range in size from 1 GiB to 1 TiB.	June 16, 2014
Amazon EBS encryption	Amazon EBS encryption offers seamless encryption of EBS data volumes and snapshots, eliminating the need to build and maintain a secure key management infrastructure. EBS encryption enables data at rest security by encryptin g your data using AWS managed keys. The encryptio n occurs on the servers that host EC2 instances, providing encryption of data as it moves between EC2 instances and EBS storage.	May 21, 2014
Incremental snapshot copies	You can now perform incremental snapshot copies.	June 11, 2013
EBS snapshot copy	You can use snapshot copies to create backups of data, to create new Amazon EBS volumes, or to create Amazon Machine Images (AMIs).	December 17, 2012