

Table of Contents

What Is Amazon EBS?	3
Understanding EBS Volume Types	3
Standard Magnetic Volumes	4
HDD Storage Volumes	4
Cold HDD (sc1) Volumes	4
Throughput-Optimized HDD (st1) Volumes	4
SSD-backed Volumes	4
gp2 General Purpose SSD Volumes	4
gp3 General Purpose SSD Volumes	4
io1 Provisioned IOPS SSD Volumes	5
io2 Provisioned IOPS SSD Volumes	5
io2 Block Express Volumes	5
EBS-Optimized EC2 Instances	5
EBS and RAID	6
RAID 0	6
RAID 1	6
Amazon EBS Multi-attach	6
EBS vs EFS	7
Understanding File Systems	8
NTFS	8
ext3	8
ext4	9
xf	9
ZFS	10
Btrfs	10
So, Why would You Care About File Systems?	12

Protecting Sensitive Data with EBS Volume Encryption	12
How EBS Pricing Works	15
Provisioned Storage Costs	15
Provisioned IOPS and Provisioned Throughput Costs	15
Snapshot Costs	16
Fast Snapshot Restore (FSR) Costs	16
EBS Direct APIs for Snapshots Costs	16
Analyzing EBS Costs	18
Monitoring EBS Volumes	18
Checking EBS Volume Status	19
Checking EBS volumes performance	19
Checking Over-Provisioned Volumes	20
EBS Bad Practice	20
Overprovisioning EBS Volumes	20
Ignoring EC2 Configuration for High-performance EBS Volumes	20
Ignoring FSR (Fast Snapshot Restore) Configuration	20
Keeping Unused EBS Volumes	20
Using a Suboptimal File System	20
Using a Single Volume to Host Everything	20
Creating Non-synchronized Snapshots	21
EBS Best Practices	21
Conclusion	23

What Is Amazon EBS?

Amazon Web Services (AWS) offers a highly performant block storage service called **Amazon EBS or Elastic Block Store**. This service enables the creation of standalone virtual hard drives in the cloud and attaching these volumes to Amazon Elastic Compute Cloud (EC2) virtual machines.

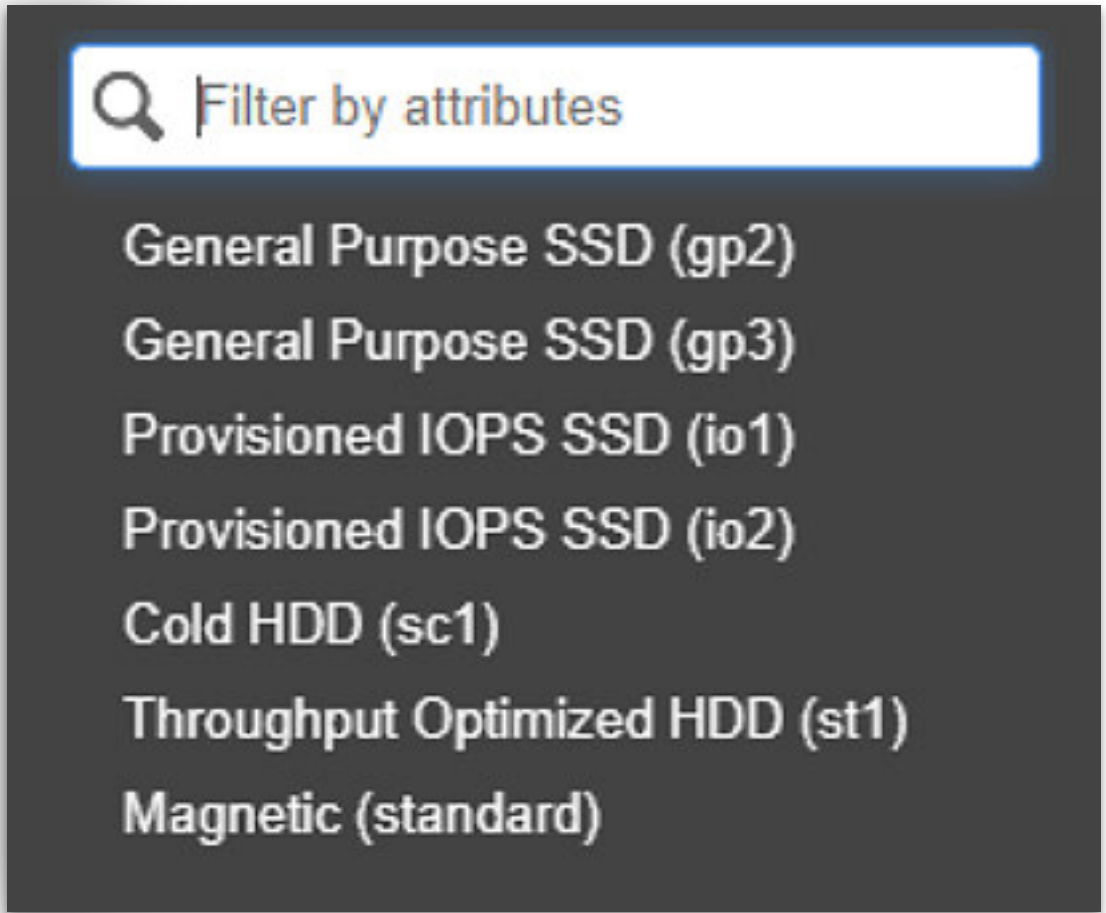
AWS customers have used EBS since its early days for almost all types of demanding workloads such as databases, applications, email, file storage, backup, or websites. EBS volumes are easy to create and configure and can be scaled to deliver extremely high IO performance. These volumes are also highly available and durable. Although EBS volumes are not replicated across multiple Availability Zones, they are copied to multiple servers in the same AZ, thus offering 99.99% availability and up to 99.999% durability. Users can also encrypt EBS volumes for data security at rest.

Understanding EBS Volume Types

Broadly, there are three types of EBS volumes:

- Previous generation standard magnetic storage volumes.
- Hard disk drive (HDD) backed storage volumes.
- Solid-State Device (SSD) backed storage volumes.

Due to the nature of the technology, SSD volumes offer the highest speed and throughput. However, many workloads are also suitable for HDD volumes.



EBS Volume Types

Standard Magnetic Volumes

The previous generation standard EBS volumes backed by magnetic hard drives are still used for some workloads today. These EBS volumes can have a maximum size of 1 TB and can offer up to 200 IOPS (Input/Output per Second). Standard magnetic EBS volumes are suitable for small-scale workloads where throughput isn't a requirement.

HDD Storage Volumes

Currently (as of May 2021), there are two types of HDD storage volumes:

Cold HDD (sc1) Volumes

sc1 type Cold HDDs are the lowest priced EBS volumes (\$0.015/GB-month) and are suitable for storing infrequently accessed data like backups and archives. These volumes generally have higher capacity and performance than the standard magnetic volumes. Cold HDDs can go up to 16 TB in size, offer a maximum of 250 IOPS (1 MB per IO), and a throughput of 250 MiB/second (262.14 MB/second)



Throughput-Optimized HDD (st1) Volumes

Next high up in the ladder is throughput-optimized st1 volumes. The significant difference between st1 and sc1 or standard magnetic volumes is that st1's offer throughput optimization. In other words, when you provision an st1 EBS volume, you can specify its IO throughput.

Like sc1s, st1 volumes can have a maximum size of 16 TB and have a 1 MB IO block size. However, the throughput can be as high as 500 IOPS per volume or 500 MiB per second. That's why st1 volumes are better suited for high throughput workloads like log management, big data, data warehouses, ETL, or streaming applications.

SSD-backed Volumes

Most write-intensive workloads are better served by SSD-backed EBS volumes. These volumes can quickly access data due to the non-rotational nature of the underlying solid-state devices (SSD). The IO block size for SSD volumes is 16 KB. At the time of writing, there are five types of SSD-backed volumes.



gp2 General Purpose SSD Volumes

gp2 is the default SSD-backed EBS volume for EC2 instances. AWS recommends using this type of volume for smaller workloads like boot volumes, non-production environments, or small-scale applications. These volumes have a baseline performance of 3 IOPS per GB of provisioned storage and can deliver up to 16,000 IOPS. The maximum volume size can be 16 TB, and the maximum throughput can be 250 MB/s.

gp3 General Purpose SSD Volumes

gp3 is the latest generation general-purpose SSD-backed EBS volume. It offers 3,000 IOPS and 125 MB/s throughput as a baseline performance for any volume size. Customers can provision up to 16,000 IOPS and 1,000 MB/s throughput for an extra fee. gp3 can be used for a wide variety of workloads that need low-latency performance such as single instance databases, VDIs, medium-scale data warehouses, and so on.

io1 Provisioned IOPS SSD Volumes

io1 provisioned IOPS SSD-backed EBS volumes offer higher IO performance, throughput, and lower latency than their general-purpose cousins. These volumes are best suited for IO and throughput-intensive workloads like MPP data warehouses, large OLTP databases, or NoSQL backends.

io1 SSD volumes can go up to 16 TB in size and can sustain a maximum IOPS of 64,000 at 1000 MB/s throughput. AWS recommends using io1 SSD EBS volumes with EBS-optimized instances for maximum performance. When attached to EBS-optimized EC2 instances, io1 SSD volumes can deliver their provisioned performance 99.9% of the time, and the latency comes down to single-digit milliseconds.



io2 Provisioned IOPS SSD Volumes

The io2 provisioned IOPS SSD EBS volumes are similar to io1 SSD volumes as they have the same maximum IOPS of 64,000 at 1000 MB/s throughput. However, it has a durability of 99.999% (io1 SSDs have 99.9%) and offers an IOPS-to-storage ratio of 500 IOPS/GB. This is 10 times more than that of io1, which has an IOPS-to-storage ratio of 50 IOPS/GB.

AWS recommends using io2 SSD volumes for workloads like SAP Hana, IBM DB2, Oracle, etc.

io2 Block Express Volumes

At the time of writing, io2 Block Express is the latest generation io2 SSD-backed EBS volume. It can go up to 64 TB in size and has a maximum IOPS of 256,000 – four times that of io1 or io2. The throughput is also four times more, with a maximum of 4,000 MB/s. The durability is 99.999%, and the IOPS-to-storage ratio is 1,000 IOPS/GB – twice that of io2.

EBS-Optimized EC2 Instances

An EBS volume's performance depends on the network capacity of the EC2 instance it's attached to. That's because EBS volumes and EC2 instances are both virtualized: EBS volumes are not physically connected to EC2 instances like traditional disks or Storage Area Networks (SAN) which are connected to physical servers via SCSI interfaces. This also means an EC2 instance uses its network channel to connect to (and share its network bandwidth with) both the outside network and its EBS volumes. For this reason, busy EC2 instances using the same network channel for both IP and storage traffic can cause significant performance bottlenecks in the attached EBS volumes.

Amazon offers a number of EC2 instance types classed as "EBS optimized". These instances use separate network channels for IP and EBS traffic. EBS volume performance in these instances is significantly higher than non-EBS optimized instances because the storage traffic doesn't have to compete for network bandwidth. As an example, General Purpose (gp2 and gp3) SSD volumes attached to EBS-optimized instances are guaranteed to deliver their baseline and burst performance 99% of the time. Provisioned IOPS (PIOPS) volumes (io1, io2) attached to EBS-optimized instances are guaranteed to deliver their provisioned performance 99.9% of the time.

The newer generation EC2 instance classes have EBS optimization enabled by default – users don't need to do anything. These instance types offer EBS bandwidth from 425 Mbps to up to a staggering 38,000 Mbps, and an EBS connection throughput from 118 MB/second to 1187.5 MB/s for 128Kb I/O chunks. The maximum IOPS supported by these instances is 173,333 for 16 KB I/O blocks.

There are other instance types that support EBS optimization, but it's not enabled by default. You need to enable it either when creating the instance, or later by modifying its properties. These instance types offer a maximum EBS bandwidth of up to 2000 Mbps, a maximum connection throughput of 2000 MB/s, and a maximum disk throughput of 16,000 IOPS. Also, you need to pay for EBS optimization once you enable it.

There are two things you should remember when choosing an EBS-optimized instance.

First, always choose a current-generation EBS-optimized instance type. These instances offer better performance than older-generation ones.

Second, ensure the throughput capacity of the instance closely matches the throughput capacity of the attached EBS volumes. For example, if an instance class has a maximum EBS channel throughput of say, 10,00 IOPS and an attached EBS volume has a maximum throughput of 6000 IOPS, the rest 4000 IOPS capacity of the EBS channel can't be used. Conversely, if you are attaching a high throughput EBS volume to a low-bandwidth EC2 instance, you will never get the full throughput performance from the disk, and you'll pay for the unused capacity.

EBS and RAID

RAID (Redundant Array of Inexpensive Disks) was (and still is) heavily used in on-premise, non-cloud environments. RAID involves configuring a group of physical disk volumes directly attached to a physical server to provide larger disk capacity and enhanced fault tolerance. [*There are different levels of RAID available.*](#) For example, RAID 5 or RAID 6 are widely used for maximum data protection.

With the adoption of the cloud, infrastructure engineers don't have to configure RAID at the hardware and network level. Instead, Amazon EBS volumes are virtualized and have their own underlying redundancy. AWS customers can also create EBS volume snapshots for data protection and provision IOPS for the required performance.

AWS still supports creating RAID volumes at a software level though. With EBS, the two common RAID options are RAID 0 and RAID 1.

RAID 0

This involves creating a RAID array with multiple EBS volumes. Like the on-premise counterpart, EBS RAID 0 does not offer any fault tolerance, but it improves write performance. With RAID 0, data is striped across all EBS volumes, improving the overall IO performance.

In addition, the storage space, IOPS, and throughput of the array is the sum of those of the individual volumes. For example, if there are three 500 GB EBS volumes in a RAID 0 array with each having 5000 IOPS and 500 MBps throughput, the resulting array will have a total of 1.5TB storage, 15,000 IOPS, and 1,500 MBps throughput.

Since there's no fault tolerance in RAID 0, losing one volume in the array will mean data loss.

RAID 1

RAID 1 involves creating an array with mirrored EBS volumes. RAID 1 can be used for instance-level data redundancy. With RAID 1, data is simultaneously written to the participating volumes – ensuring data is still available if one mirrored volume is lost. Also, the total disk space, IOPS, and throughput are the same as those of the individual participating volumes.

AWS doesn't recommend creating RAID 5 or RAID 6 with EBS. That's because writing the parity bit across all volumes can cause a loss of the available IOPS.

Creating snapshots of EBS volumes in RAID configuration can be tricky. With RAID 0, data integrity can't be guaranteed if the volume snapshots are not synchronized. That's why AWS recommends creating [*EBS multi-volume snapshots*](#). With multi-volume snapshots, it's possible to create simultaneous, consistent snapshots of up to 40 volumes attached to an EC2 instance.

Amazon EBS Multi-attach

For workloads requiring extremely high performance, multiple EC2 instances may need to access the same storage space simultaneously. This is usually the case for clustered applications. EBS provides an advanced feature called "multi-attach" to cater to this use case, but it has several caveats.

With EBS multi-attach, an io1 or io2 EBS volume can be attached to up to 16 Nitro Linux instances in the same Availability Zone in the same region. This feature isn't available in all regions, and it may limit other EBS functionalities like resizing a volume.

The multi-attached common volume needs to be formatted as a clustered file system, and it doesn't support I/O fencing. Regular filesystems like ext4 don't support this type of volume.

As this is an advanced setup for special use cases, make sure you understand the trade-offs and your OS, file system, and application support the feature. For a complete list of limitations in multi-attached EBS volumes, check the [*AWS documentation*](#).

EBS vs EFS

AWS offers another storage service called the Elastic File System (EFS). EFS is a managed Network File System (NFS) designed for Linux-based EC2 instances and on-premise servers. There's a similar storage system for Windows hosts called the [Amazon FSx for Windows File Server](#). FSx uses the Server Message Block (SMB) protocol while EFS uses NFS.

Like EBS, EFS also offers high durability. However, the main difference lies in scalability. EFS volumes can scale up quickly and automatically to meet abrupt spikes in workload demand and scale down with a decreased load. This makes EFS more flexible than EBS. Also, unlike EBS, where maximum volume size can be up to 16 TB, there's no maximum storage size for EFS volumes.

This scalability also means EFS volumes don't need to be pre-provisioned with a specific size for an anticipated load, which ultimately saves costs. Similar to EBS, you can also specify a provisioned throughput for EFS volumes.

Performance mode

Set your file system's performance mode based on IOPS required. [Learn more](#)

☒ General Purpose

Ideal for latency-sensitive use cases, like web serving environments and content management systems

☐ Max I/O

Scale to higher levels of aggregate throughput and operations per second

Throughput mode

Set how your file system's throughput limits are determined. [Learn more](#)

☐ Bursting

Throughput scales with file system size

☒ Provisioned

Throughput fixed at specified amount

Provisioned Throughput (MiB/s)

Add throughput value

Valid range is 1-1024 MiB/s

Maximum Read Throughput (MiB/s)

Creating EFS Volume

EFS also offers [lifecycle management](#), a price-saving feature similar to S3 lifecycle management. EFS lifecycle management enables the automatic and transparent transfer of infrequently accessed data to a separate storage class.

Lifecycle management

Automatically save money as access patterns change by moving files into the EFS Infrequent Access storage class. [Learn more](#)

30 days since last access

EFS Lifecycle Management

Another difference between EBS and EFS is that an EFS volume can be mounted on hundreds or even thousands of nodes, whereas generally, an EBS volume can be attached to a single EC2 node only. As we noted earlier however, EBS multi-attach is the exception to this rule.

This EFS feature can be quite cost effective. . EFS volumes cost more than EBS volumes, but mounting an EFS volume to multiple EC2 instances will have the same cost as mounting it to a single instance. In comparison, creating and attaching EBS volumes for every node will quickly add to the bill.

So, should you use EFS for all your EC2 instances instead of EBS? Not at all. EBS volumes are best suited for relational and NoSQL databases, enterprise applications like ERP systems, mail servers, SharePoint, web servers, directory servers, DNS servers, or middlewares. That's because these systems typically don't run on large clusters, and therefore don't need a commonly mounted volume.

The performance requirements of these workloads can also be met by existing EBS volume types. EBS volume snapshots are sharable with other EC2 instances in different regions. Also, EBS volumes can be attached to both Windows and non-Windows EC2 machines whereas EFS volumes are designed for Linux-based hosts only.

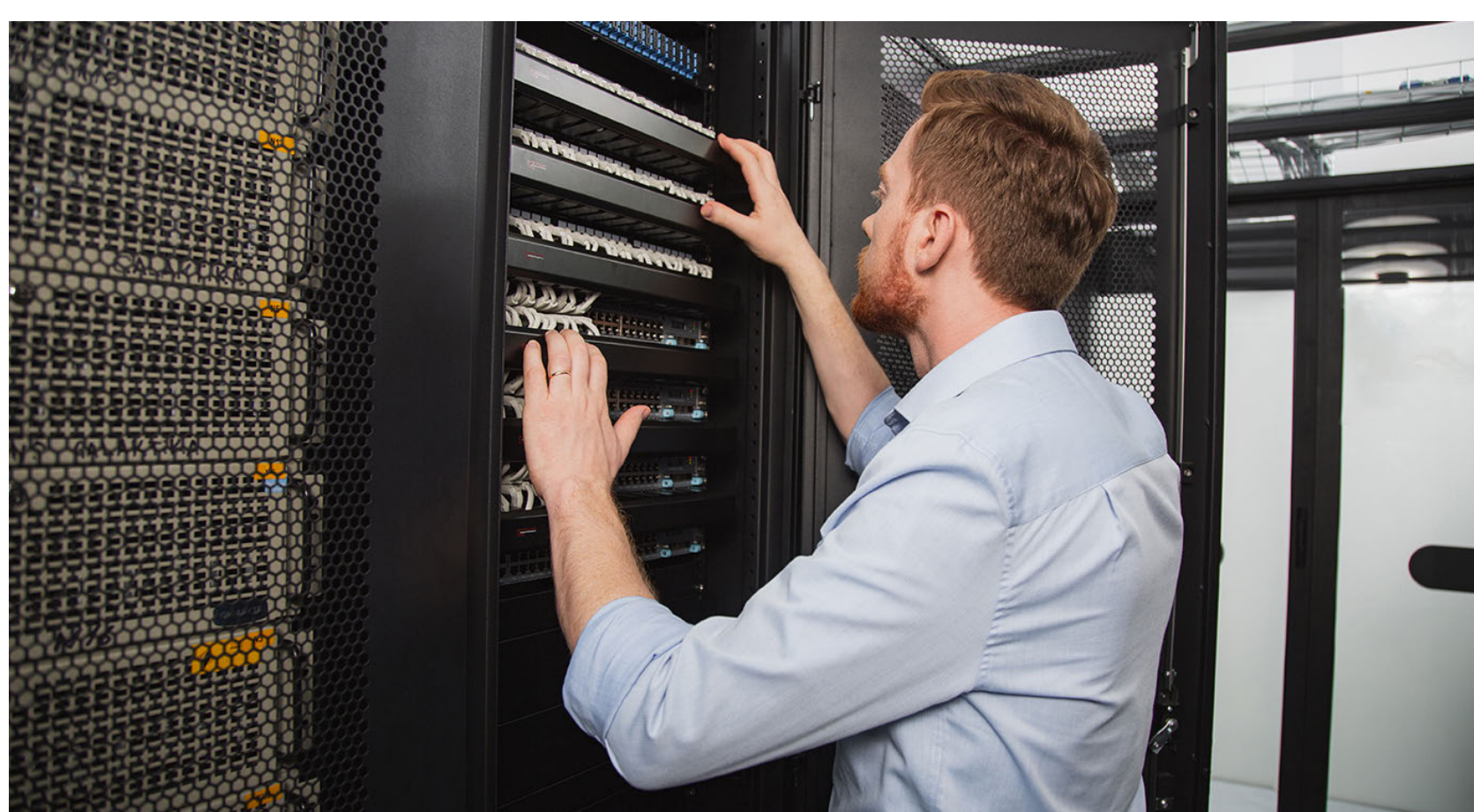
EFS volumes are best suited for enterprise-wide file servers, backup systems, Big Data clusters, Massively Parallel Processing (MPP) systems, Content Distribution Networks (CDN), and other such large use cases.

Understanding File Systems

A file system controls how data in a computer storage medium is stored, arranged, addressed, and retrieved. It also controls the fault tolerance and compression of the data and stores its different attributes. The computer's operating system offers commands for interacting with the underlying file system. This includes both high-level operations like copy, move, delete, or create directories, and low-level operations like create partitions and formatting.

There are many different file systems available for different types of storage mediums like hard disk drives, removable SSD drives, tape or optical drives, NFS volumes, or even temporary mediums like RAM disks. For most computing needs though, direct-attached storage is still the mainstay today.

Since EBS can be logically classed as direct-attached storage, we will talk about the major file systems you can use with EBS volumes. These include both Linux and Windows file systems.



NTFS

New Technology File System (NTFS) is Microsoft's proprietary file system used in the Windows family of operating systems including Windows Servers. It superseded the older FAT and FAT32 file systems.

NTFS offers features like:

- Journaling for crash protection of files.
- Volume Shadow Copy Service to keep older versions of files and folders using copy-on-write technology. This allows users to recover older versions of files and folders.
- Fine-grained security through Access Control Lists (ACLs).
- Enforceable space quotas for users.
- File encryption, compression, and sparse files.
- Ability to expand or shrink partitions.

According to Microsoft, NTFS can support up to 8 Petabytes of maximum file size and volume size. With the default 4KB page size (i.e., the data block size for an IO operation) though, it can support up to 16 TB file and partition size, which is also the maximum volume size supported by EBS.

ext3

The third Extended File System (ext3) comes with most Linux distributions. It offers journaling for crash protection. The maximum block size for file IO is 8 KB in ext3, which is less than what most modern file systems offer. Also, ext3 lacks some critical features like online defragmentation, undelete, or compression.

ext4

The fourth Extended File System (ext4) is an improved version after ext2 and ext3. At the time of writing, most Linux distributions' root volume comes with the ext4 file system by default.

Its improvements over ext3 include:

- Use of extents to facilitate block allocation of large files.
- Online defragmentation.
- Faster file system consistency checks.
- Delayed allocation of blocks to files (only when they are written).
- Unlimited number of subdirectories.
- Support of maximum 1 Exbibyte volume size and 16 TB file size.

xfs

Also known as Extended File System, xfs is a 64-bit journaled file system with features similar to ext4. XFS is the default file system for Red Hat Enterprise Linux.

Some of its features are:

- Metadata journaling for crash recovery.
- Online defragmentation and enlargement of the file system.
- Delayed block allocation to files.
- B-trees for file and directory allocation.
- Guaranteed Rate I/O (GRIO). This allows applications to reserve bandwidth.

xfs offers a maximum file size and volume size of 8 Exbibytes.

ZFS

The Zettabyte File System (ZFS) was originally developed by Sun Microsystems and later ported to FreeBSD. There are two variants of ZFS today: the original ZFS – now owned by Oracle Corporation, and closed source, and the Open ZFS, still offered via Common Development and Distribution License. Due to CDDL’s incompatibility with GNU Public License (GPL), most Linux distros don’t have it as part of their kernel, although they offer methods to install it.

This is an advanced file system. In fact, when it was first developed, it was meant to be the ultimate file system.

Some of ZFS’ features include:

- Ability to span the file system across a pool of drives such as a volume manager, and continue to add more.
- Copy-on-write capability. This helps ZFS to recover older versions of overwritten files and directories.
- Automatic snapshot of the entire file system for change tracking of the data, and rollback capability for recovery purposes.
- A maximum of 16 Exbibyte file size and a maximum of 256 Quadrillion Zettabytes volume size.

To use ZFS out-of-box on EBS volumes, it’s best to use a [FreeBSD AMI](#). There’s also the option to install ZFS in Linux. You can check out the [ZFS on Linux site](#) to learn more.

Btrfs

Btrfs (often pronounced “ButterFS”, “Better FS”, or “Btree FS”) was also originally developed by Oracle Corporation and later merged into the main Linux kernel. Several vendors including Red Hat, Intel, and SuSE are now contributing to its development. Btrfs uses B-tree structures for data storage and retrieval.

It offers features like:

- capCopy-on-writeability.
- Logical volume management capability (like ZFS).
- Readable and writable snapshots of the entire file system that can be accessed like a regular folder.
- Built-in support for RAID 0, RAID 1, and RAID 10.
- Automatic Cyclic Redundancy Check (CRC) to ensure the integrity of data and metadata.
- Transparent compression with both zlib and LZO.
- Efficient storage of files smaller than the default block size.
- Online defragmentation.
- Maximum file size and volume size of 16 Exbibytes.

The following table shows a quick comparison between the file systems using a 4KB block size:

File System	OS Support	Maximum Volume Size	Maximum File Size	Maximum Number of Files	Deduplication	Online Size Extension	Online Size Shrink
NTFS	Microsoft Windows & Windows Server	8 PB	8 PB	$2^{32} - 1$	Yes	Yes	Yes
ext3	Linux	16 TB	2 TB	Minimum between (volume size / 2^{13}), & the number of blocks)	No	Yes	No
ext4	Linux	1 EiB	16 TiB	$2^{32} - 1$	No	Yes	No
xfs	Linux	8 EiB	8 EiB	2^{64}	Yes	Yes	No
ZFS	FreeBSD, Open Solaris, Linux	256x2^50 ZiB	16 EiB	2^{48} in any directory	Yes	Yes	Yes
Btrfs	Linux	16 EiB	16 EiB	2^{64}	Yes	Yes	Yes

File System	OS Support	Copy On Write	Checksum	Encryption	Compression
NTFS	Microsoft Windows & Windows Server	Yes	No	Yes	Yes
ext3	Linux	No	No	No	No
ext4	Linux	No	No	Yes	No
xfs	Linux	Yes	No	No	No
ZFS	FreeBSD, Open Solaris, Linux	Yes	Yes	Yes	Yes
Btrfs	Linux	Yes	Yes	No	Yes

So, Why would You Care About File Systems?

The EC2 instance's root volume comes pre-installed with the operating system's default file system. However, you can choose the file system of the EBS volumes you attach to the machine. Depending on the type of workload, the file system you choose can have a wide impact on performance and fault tolerance.

For Windows Servers, the choice of a file system is fairly straightforward: it's NTFS. However, you may want to enable the Volume Shadow Copy or the encryption service for NTFS volumes storing critical data. Similarly, choosing a file block size equal to the database page size can increase performance as the database engine doesn't have to wait for multiple IO hops from the operating system.

For Linux systems, features like copy-on-write, compression, or encryption will dictate the choice of the file system. If your server is supporting large-scale databases, busy web servers, or critical middleware, you may want to consider features like online file system extension, or multi-volume management. If fault-tolerance is a priority (e.g., for backup servers), you would want a file system with a transparent snapshot facility.

Although storage is cheap, large savings in both space and cost can often be achieved with data deduplication. When deduplication is enabled, the file system examines areas in the volume where the same data has been written multiple times. It then stores that data only once – optionally compressing it – and ensures it can be reproduced for each file that contains it. Data deduplication can be used for large backup volumes, database snapshots, etc.

Choosing the right file systems for the OS, swap space, application, and data should therefore be a part of your application architecture process.

Here's a quick checklist:

- Btrfs is suitable for high fault tolerance.
- ZFS is suitable for extremely large computing needs (e.g. dealing with astronomical data, very large-scale graphics manipulation), or running Oracle applications.
- ext4 can be used for a wide variety of vanilla data storage requirements.
- xfs can be very efficient for storing and manipulating large files.

Protecting Sensitive Data with EBS Volume Encryption

Organizations often need to implement encryption in order to protect their data both at rest and in transit. Sometimes this is a requirement dictated by the sensitive nature of the data or by industry or region-specific regulations. Amazon makes it easy to encrypt EBS volumes for data protection.

An encrypted EBS volume ensures data inside it is encrypted, and all data moving between the volume and the EC2 instance is also encrypted. Any snapshot created from an encrypted volume is also encrypted, and so are all new volumes created from that snapshot.

You can use Amazon Key Management Service (KMS) to encrypt an EBS volume.



The high-level steps for EBS encryption and decryption works like this:

- You choose the option to encrypt an EBS volume when you launch an EC2 instance or create it from the EBS console. The image below shows EBS encryption when creating a volume.

Volumes > Create Volume

Create Volume

Volume Type

General Purpose SSD (gp2)

Size (GiB)

100

(Min: 1 GiB, Max: 16384 GiB)

IOPS

300 / 3000

(Baseline of 3 IOPS per GiB with a minimum of 100 IOPS, burstable to 3000 IOPS)

Throughput (MB/s)

Not applicable

Availability Zone*

us-east-1a

Snapshot ID

Select a snapshot

Encryption

☒ Encrypt this volume

Master Key

(default) aws/ebs

KMS Key Description

Default master key that protects my EBS volumes when no other key is defined

KMS Key Account

This account ()

KMS Key ID

75e0e6ec0ce8

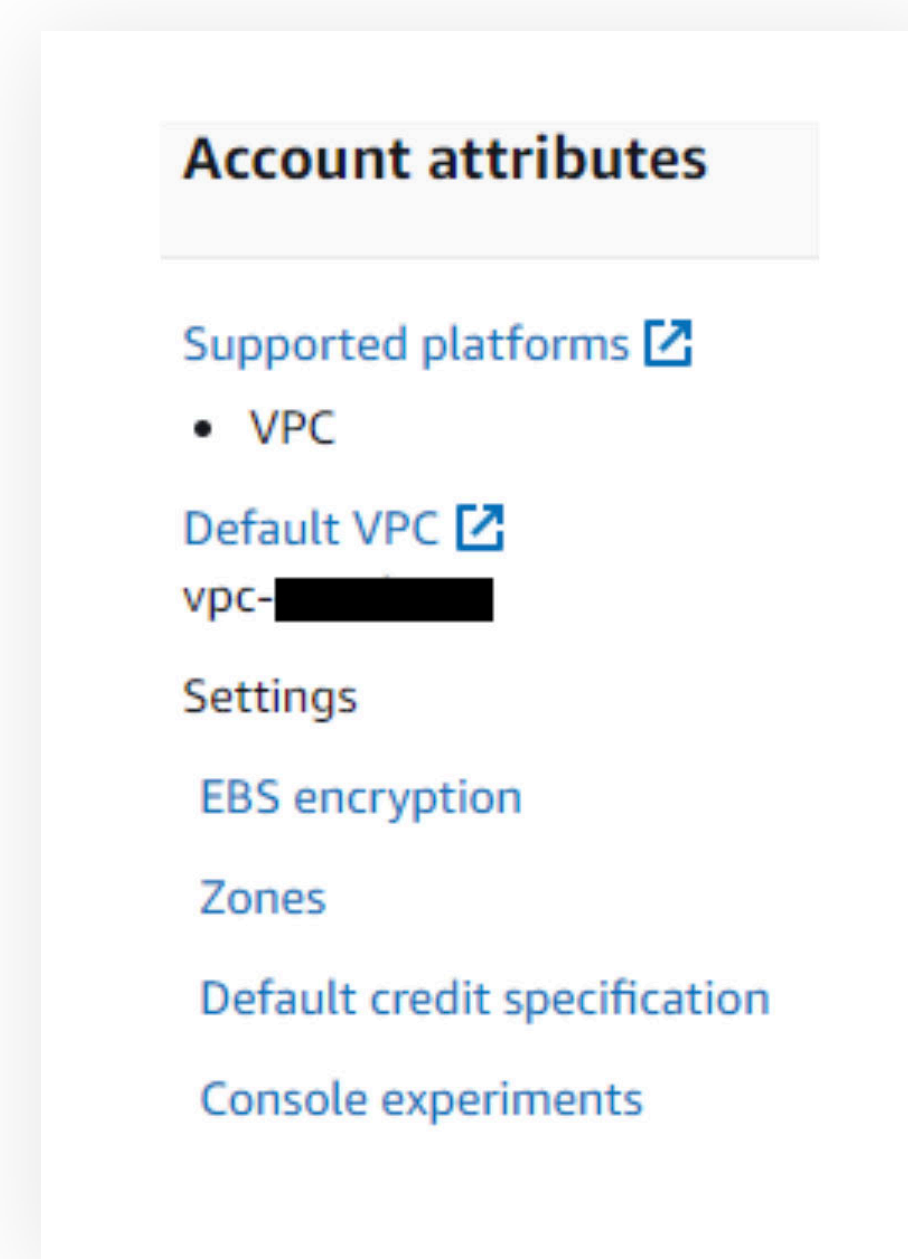
KMS Key ARN

arn:aws:kms:us-east-1: :key:75e0e6ec0ce8

Creating Encrypted EBS Volume

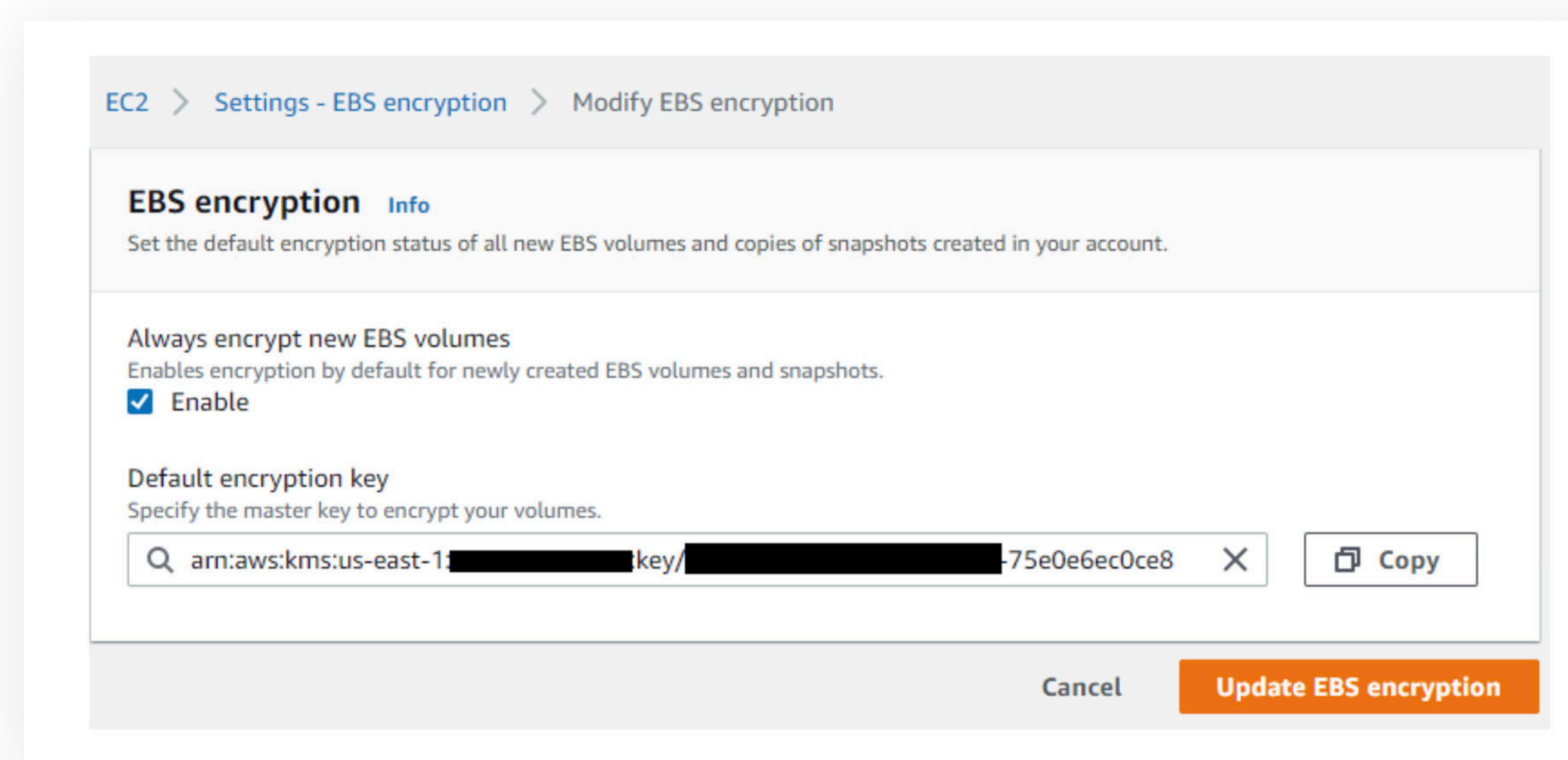
- The Elastic Block Store Service calls the Key Management Service to create a Data Encryption Key or DEK. The DEK is a symmetric key using the AES-256 algorithm.
- KMS generates a DEK and encrypts it with a key called the Customer Master Key (CMK). Now, you may already have a custom CMK in the KMS, or it can be the default CMK for EBS in that region. In the image above, we are using the default CMK called `aws/ebs`.
- KMS provides the encrypted DEK to the EBS service which then stores the key in the disk.
- The encrypted EBS volume is attached to an EC2 instance.
- When the EBS volume is attached, the EC2 instance contacts the KMS with a decrypt request for the encrypted DEK stored on the disk.
- KMS provides the unencrypted DEK to the EC2 instance which then loads it into the hypervisor memory. The decrypted key is stored in the memory as long as the EBS volume is attached to the instance.
- The EC2 instance uses the DEK to encrypt and decrypt data it stores in the volume or reads from it.

The CMK we used in this example is the one EBS creates in each region and uses as the default key. You can also configure EBS to use a separate default CMK for all encryptions. To configure this, go to the EC2 console and select "EBS encryption" from the "Account attributes" panel:



EC2 Dashboard's Account Attributes Panel

Once the "Settings – EBS encryption" screen comes up, you can modify it, and select the default encryption CMK you want to use in future:



Notice the checkbox to enable "Always encrypt new EBS volumes". When you enable this option, all new EBS volumes you create will be encrypted. The snapshots you create from unencrypted volumes will be encrypted, and the volumes you restore from those snapshots will also be encrypted.

How EBS Pricing Works

Provisioned Storage Costs

An EBS volume's storage size is charged as GB-month. That's the total size of the volume in GB over a total month, calculated per second increments, with a 60-second minimum.

So, let's say you provisioned a 500 GB gp2 General Purpose SSD volume for your EC2 instance running in Mumbai, India. The storage cost in that region at the time of writing is \$0.114 per GB-month of provisioned storage. Let's also say, you deployed the volume on the 14th of September at 2:00 PM.

With 30 days in September, it would be 15 days and 10 hours (2 PM to 12:00 AM on 14 September) before the current calendar month's billing period ends for that EBS volume.

In other words, the EBS volumes will be up for:

- $(10 \times 3,600) \text{ seconds} + (15 \times 24 \times 3,600) \text{ seconds} = 1,332,000 \text{ seconds}$.
- In September we have $(30 \times 24 \times 3600) = 2,592,000 \text{ seconds}$.
- So, the storage charge will be $500 \text{ GB} \times \$0.114/\text{GB-month} \times (1,332,000 \text{ seconds} / 2,592,000 \text{ seconds}) = \29.29 .

EBS volume costs depend on a few factors. These are:

- The volume type and the storage size.
- The volume's provisioned throughput.
- The volume's provisioned IOPS.
- The volume's total snapshot size.
- Fast Snapshot Restore (FSR).
- EBS direct APIs for snapshots.

Not all the costs apply to all volume types, and sometimes you won't be using some of the features like FSR. Also, there are free tiers for baseline performance and throughput.

Provisioned IOPS and Provisioned Throughput Costs

If you decided to deploy the EBS volume as gp3 General Purpose SSD, you could provision both its throughput and IOPS. These charges are not applicable to the SSD gp2 volume you provisioned.

The calculation for provisioned throughput (MB/s) and provisioned IOPS work the same way as GB-month. Provisioned throughput charge is calculated as MB/second-month, and provisioned IOPS is charged at IOPS-month.

So let's say, instead of gp2, you deployed the volume as gp3. You also specified a provisioned IOPS of 15,000, and a disk throughput of 500MB/second.

Currently, Amazon doesn't charge anything for up to 3,000 IOPS and 12 MB/second. So your AWS account will only be charged for $15,000 - 3,000 = 12,000 \text{ IOPS}$, and $500 \text{ MB/s} - 125 \text{ MB/s} = 375 \text{ MB/s}$ for the volume.

Also, at the time of writing, in the AWS Mumbai region, gp3 volumes cost:

- \$0.0912/GB-month for storage.
- \$0.0057/provisioned IOPS-month above 3,000 IOPS.
- \$0.0456/provisioned MB/s-month over 125 MB/second provisioned.

Using all this information, let's calculate the total charge for the gp3 volume.

The storage cost will be $500 \text{ GB} \times \$0.0912/\text{GB-month} \times (1,332,000 \text{ seconds} / 2,592,000 \text{ seconds}) = \23.43 .

The provisioned IOPS charge will be $12,000 \text{ IOPS} \times \$0.0057/\text{provisioned IOPS-month} \times (1,332,000 \text{ seconds} / 2,592,000 \text{ seconds}) = \35.15

The provisioned throughput charge will be $375 \text{ MB/second} \times \$0.0456/\text{provisioned MB/s-month} \times (1,332,000 \text{ seconds} / 2,592,000 \text{ seconds}) = \8.79

The total charge for that volume in September would be: $\$23.43 + \$35.15 + \$8.79 = \67.37 .

Snapshot Costs

Another EBS price component relates to snapshots. EBS snapshots are saved in S3 in a compressed format. The first snapshot contains all the contents of the volume, and subsequent snapshots only contain contents of the changed block. This also means that your AWS account will be charged only for the changed blocks after the first snapshot.

At the time of writing, EBS snapshots are charged \$0.05 per GB-month of data stored.

Other factors that affect snapshot cost include:

- Snapshot frequency.
- The compression ratio of the data.
- Data transfer costs for cross-region snapshots.

Fast Snapshot Restore (FSR) Costs

Amazon Fast Snapshot Restore (FSR) allows restoring an EBS snapshot without the initialization overhead at volume creation time. This eliminates the usual I/O latency when a block in the restored volume is accessed for the first time. With FSR, a restored EBS volume immediately starts operating with its provisioned performance.

FSRs are charged in Data Services Unit-Hours (DSUs) for each snapshot and each Availability Zone (AZ) where it's enabled. DSUs are charged per minute with a minimum of 1 hour. You can enable FSR when you create a snapshot. AWS will keep charging the FSR-enabled snapshot until you disable it.

FSR costs \$0.75 per DSU hour in each Availability Zone where it's enabled.

EBS Direct APIs for Snapshots Costs

Amazon Direct APIs for snapshots allow applications to directly read data from snapshots and find differences between two snapshots. This feature is mainly meant for backup applications, in order that they can find the changes between snapshots and reduce the backup time.

At the time of writing, Amazon lists three API pricing:

- ListChangedBlocks and ListSnapshotBlocks API calls are charged \$0.0006 / 1,000 requests.
- GetSnapshotBlock API call is charged \$0.003 / 1,000 SnapshotAPIUnits.
- PutSnapshotBlock API call is charged \$0.006 / 1,000 SnapshotAPIUnits.

Amazon prices change all the time and EBS volume types also undergo changes, so it's best you refer to the [EBS pricing page](#) for the latest price. As a general rule though, you can refer to the following table for applicable charges across different volume types:

	Storage	Provisioned IOPS	Provisioned Throughput	EBS Snapshots	Fast Snapshot Restore	EBS Direct APIs for Snapshots
General Purpose SSD (gp2)	×			×	×	×
General Purpose SSD (gp3)	×	×	×	×	×	×
Provisioned IOPS SSD (i01)	×	×		×	×	×
Provisioned IOPS SSD (i02)	×	×		×	×	×
Throughput Optimized HDD (st1)	×			×	×	×
Cold HDD (sc1)	×			×	×	×


You can also refer to the [previous generation EBS pricing page](#) for magnetic volume pricing.

Magnetic volumes are priced by:

- Provisioned storage per GB-month.
- Per 1-million I/O requests.

The easiest way to calculate EBS costs is to access the [AWS pricing calculator](#). In the image below, we are using the same figures for the 500 GB gp3 volume in the us-east-1 region and calculating the costs for a full month (730 hours) and two daily snapshots. The snapshots will have 200 MB of data changes per snapshot:

Service Settings [Info](#)



Calculating EBS snapshots

[Learn more](#) on how EBS snapshot prices are calculated.

Number of instances

1

Average duration each instance runs

730

hours per month

▼

Storage for each EC2 instance

Choose EBS volume storage type.

General Purpose SSD (gp3)

▼

General Purpose SSD (gp3) - IOPS

gp3 supports a max of 16,000 IOPS per volume

15000

General Purpose SSD (gp3) - Throughput

gp3 supports a max of 1000 MBps per volume

500

MBps

▼

Storage amount

500

GB

▼

Snapshot Frequency

2x Daily

▼

Amount changed per snapshot

200

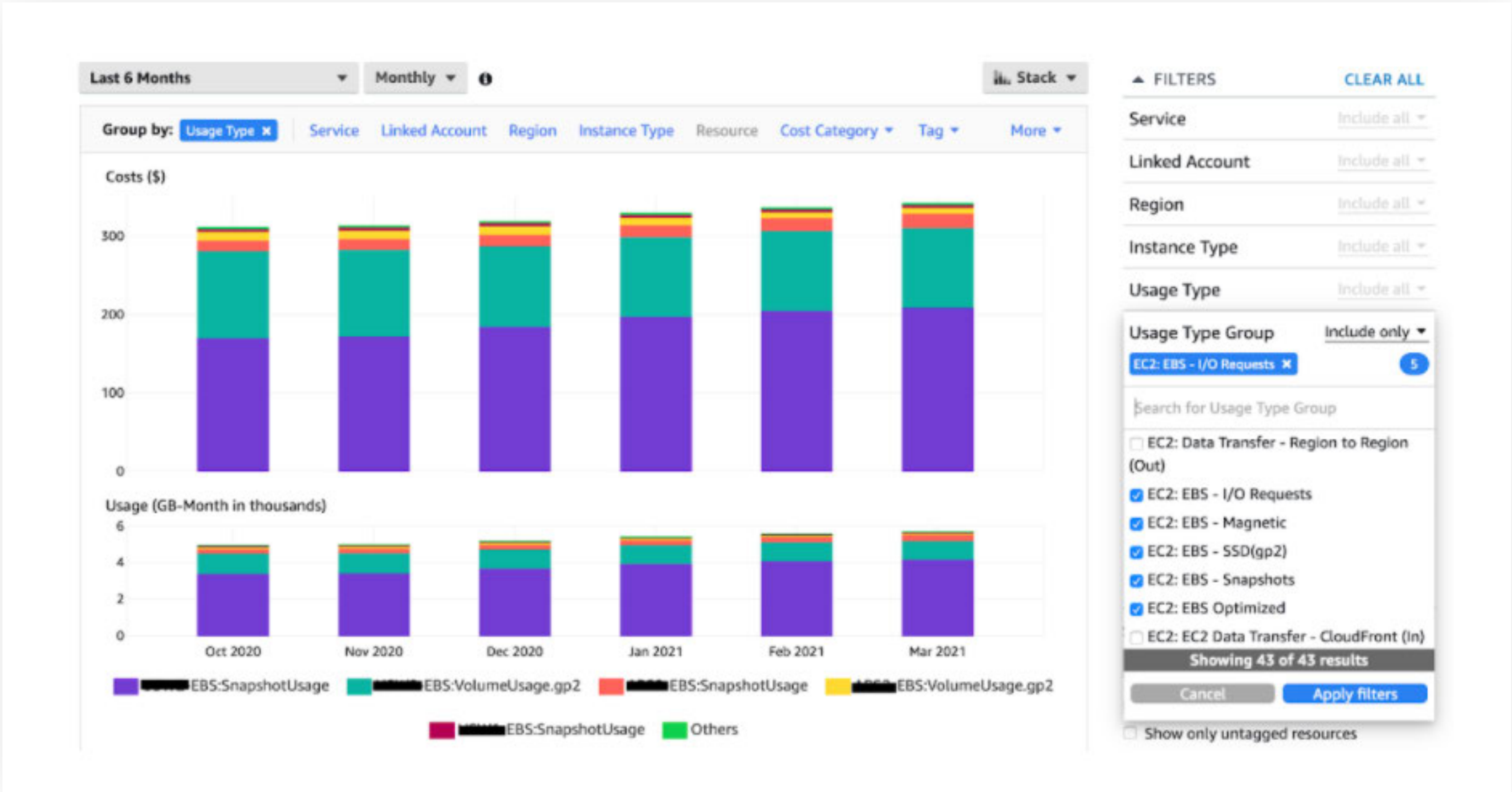
MB

▼

Calculating EBS Cost

Analyzing EBS Costs

You can use the AWS Cost Explorer to check the EBS costs incurred in your account. Just select EBS groups in the “Usage Type Group” dropdown list, and you can analyze EBS costs by region, AWS account, or usage type.



AWS Cost Explorer

As with most resources in AWS, using meaningful and consistent tagging of EBS volumes and snapshots will allow you to further drill down and apportion the costs.

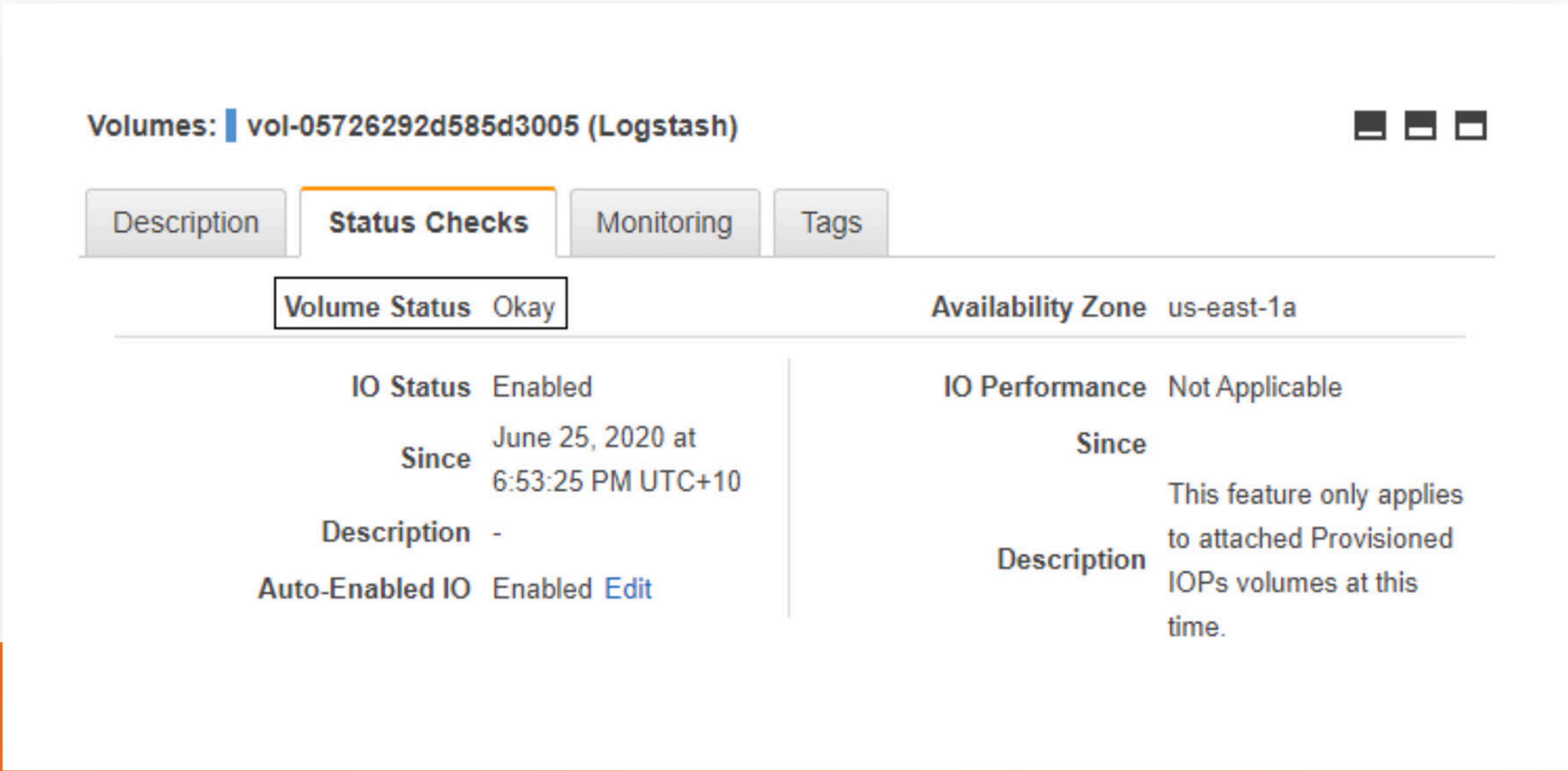
Monitoring EBS Volumes

Amazon EBS-related status and metrics can be captured from the EBS console and CloudWatch. This information shows the volume status, throughput, IO performance, and can help detect saturation and bottlenecks.

Checking EBS Volume Status

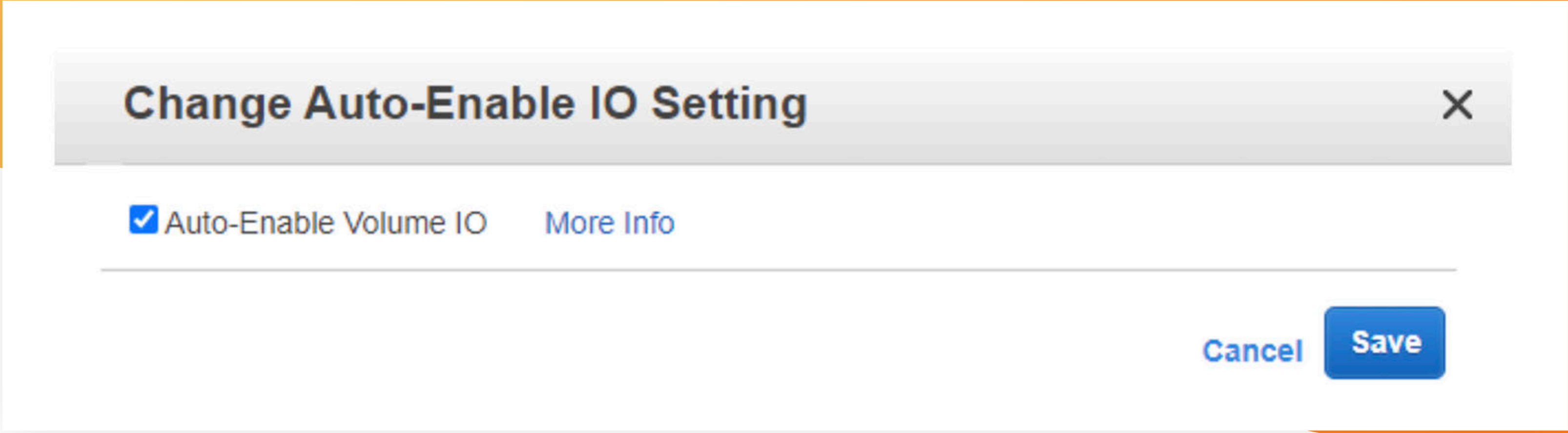
AWS will run automatic checks to determine the status of EBS volumes. This can be checked from the AWS console.

All volumes will have a “Volume Status”. io1 and io2 disks will also have an extra check for “IO Performance”, which shows a comparison between the current volume performance against its expected performance.



EBS Volume Status

A status of “Okay” means the volume is working as expected. Unhealthy volumes are marked as “Impaired”. Impaired volumes will automatically have I/O operations disabled to prevent data corruption and file system inconsistencies but this behavior can be overridden by setting the Auto-Enabled IO attribute for the EBS volume.



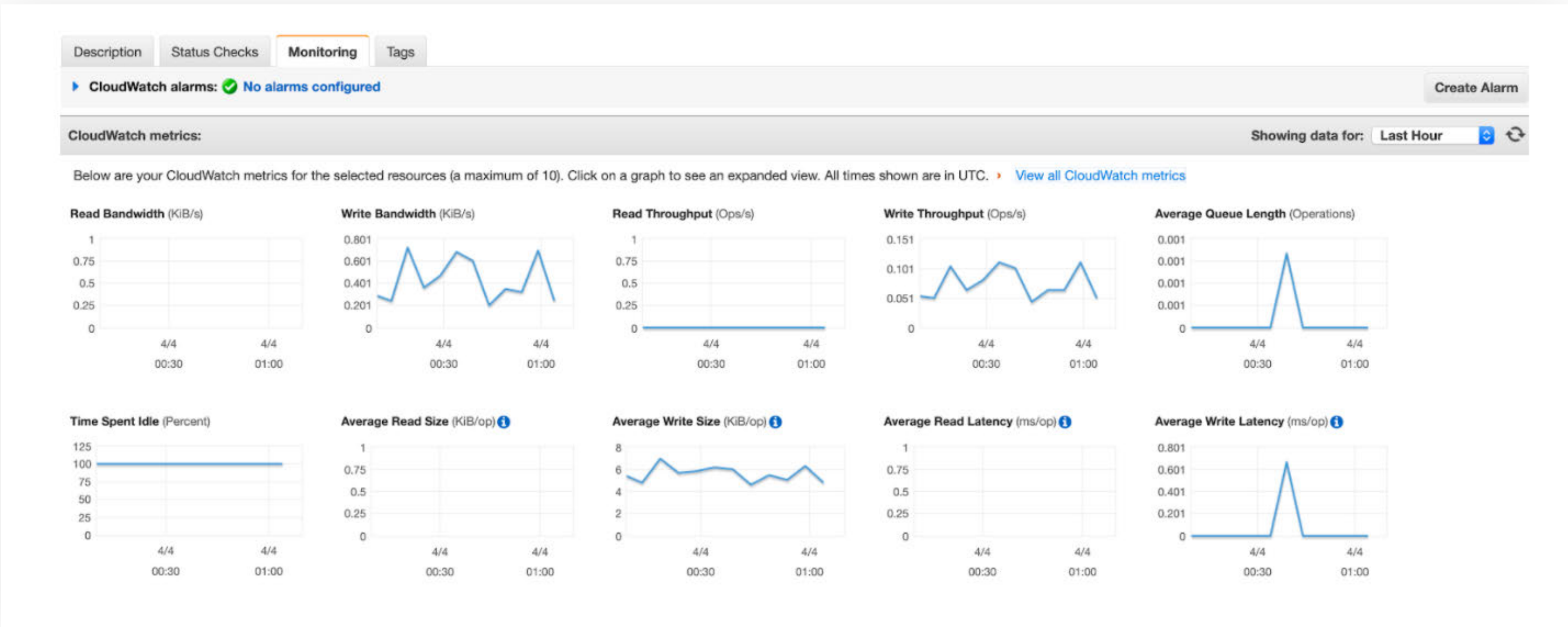
Auto-Enable Volume IO

Although it’s rare to see an impaired EBS volume, if it happens, administrators need to stop any applications using the impaired volume, re-enable I/O for the volume in the AWS console, log into the EC2 instance, and then run a filesystem repair tool to check for disk errors.

Checking EBS volumes performance

Inadequately provisioned IOPS can cause EBS volumes to experience throttled IO operations, low burst balance, maximum throughput saturation, or even EC2 network bandwidth saturation.

To check possible IOPS limitations, start by looking at the VolumeQueueLength metric. It shows the number of IO requests waiting to be completed. A consistently high value for this metric can mean throttling. You can also check the bandwidth and throughput-related metrics, including Volume *ThroughputPercentage*.



EBS Performance Charts

Some volumes are “burstable” (e.g., gp2, st1, and sc1), and can accumulate “performance credits”. This means even if the volume doesn’t use a lot of resources, it has the ability to perform above baseline for all the credits it accumulated over time. If your volume is regularly running low in burstable credit balance, it should be resized or changed to a provisioned IOPS type.

If all metrics look okay, make sure EBS volumes are not reaching their maximum throughput limit or that the EC2 instance isn’t reaching its network bandwidth limit.

Checking Over-Provisioned Volumes

It's important to keep an eye on over-provisioned volumes, particularly those with provisioned IOPS, as it can quickly become costly.

Metrics to watch for provisioned IOPS volumes include `VolumeThroughputPercentage` – which shows the percentage of total provisioned IOPS delivered, `VolumeQueueLength`, and `VolumeIdleTime`. Overprovisioned disks should have a low `VolumeThroughputPercentage` and `VolumeQueueLength`, and may have a high `VolumeIdleTime`.

EBS Bad Practices

Many organizations don't provision or use EBS volumes in an optimal way that can offer the best possible price-performance ratio. There are few common “bad practices”.

Overprovisioning EBS Volumes

The most common EBS bad practice is over-provisioning. Overprovisioning happens when large-capacity, highly performant EBS volumes are attached to EC2 instances but contain a very small amount of data, or work well above the required performance. Typically overprovisioning happens when volumes are created without any clear application architecture or planning about capacity and performance requirements. Overprovisioning results in unnecessary costs.

Ignoring FSR (Fast Snapshot Restore) Configuration

By default, after creating an EBS volume from a snapshot, the data isn't readily available inside the volume. The data is lazily loaded only when it's accessed.

For certain workloads like databases, this can mean the volume performs slower than desired for a long time until all the data has been accessed at least once.

Fast Snapshot Restore addresses this issue. Although it comes with extra costs, it should be carefully considered for latency-sensitive workloads.

Using a Suboptimal File System

A file system dictates how data in a storage volume is stored, accessed, searched, written, or tracked. There are different file systems for different operating systems like Linux, UNIX, or Windows.

One file system often performs better than another for the same workload type. Often an EC2 instance is running multiple types of workloads, each accessing a different volume for its data. It then becomes a question of using different file systems for different EBS volumes attached to the EC2 instance. More often than not, the same file system is used in all the volumes – resulting in suboptimal performance.

Ignoring EC2 Configuration for High-performance EBS Volumes

The performance of an EBS volume is highly dependent on the network bandwidth and throughput of the EC2 instance it's attached to.

EBS volumes are accessed via a network, that's why attaching a highly performant EBS volume to a low network throughput EC2 instance will result in low volume utilization. Using a high-performance EBS volume with a non-EBS-optimized EC2 will also result in bigger latency.

Keeping Unused EBS Volumes

This is another bad practice where EBS volumes are provisioned for possible future use or detached from existing EC2 instances, but never used. These unattached volumes sit idle in the customer's account, costing money despite the fact that they are not being used.

Using a Single Volume to Host Everything

This bad practice stems from how easy it is to provision EC2 instances, attach EBS volumes, and not give enough thought to the workload.

In this model, the operating system, application, data, logs, and swap space all share the same volume. It's often argued that using a single disk for everything was an issue in the old days when hard drives were directly attached to physical servers. Today, indeed a cloud-hosted volume's storage space maps to multiple, geographically separated machines. However, this is more to do with capacity than performance.

As far as the operating system is concerned, it still sees the volume as a storage area with a finite limit. So when the volume runs out of space, the applications in the machine will still be affected.

Take the example of a single-volume EC2 instance running a critical database. This machine will have an operating system cache, log files, and temporary files – all generated in the same volume where the data files are stored. If the log files are all blown up in size (perhaps due to an application bug sending huge trace messages to the files), the storage may run out far quicker than anticipated. This condition would affect the database's availability.

The only time using a single volume or the root volume to store everything makes sense is when the machine is hosting a small or less critical application. It also makes it easy to snapshot the volume.

Creating Non-synchronized Snapshots

Snapshots are used to backup EBS volumes. The snapshot process is usually automated: a scheduled task runs a backup program or script against every EBS volume attached to every EC2 instance. Once completed successfully, the job usually sends a message to the operations team. It also sends a warning if it can't snapshot one or more volumes.

Logically, this is a perfect solution. In reality, many snapshots need precise synchronization. For example, let's consider an application that runs on two EC2 servers. One EC2 machine hosts the app's database files, the other one hosts its binaries, logs, and configuration files. Throughout the day, both configuration files and the database are updated, and both need to be in sync.

Now imagine this. The automated snapshot process backs up the database volume at the start of its job, and after a few hours, snapshots the configuration files' volume. Between the two snapshots, there's a gap of a few hours – making the volume backups out-of-sync. When restored, these volumes may not work in sync for the application to come online properly.

EBS Best Practices

Here is a quick list of best practices for AWS EBS volumes. We have intentionally kept it short so you can use it as a checklist.

Area	Recommended Best Practices
Storage Setup	<p>Use individual EBS volumes for different components of a system running in the EC2 instance – based on their performance requirements, overall cost optimization, and availability needs. For example, you can provision:</p> <ul style="list-style-type: none">◦ The instance root volume for the OS, which is the default.◦ A dedicated smaller volume of gp2 or gp3 type for swap space or the Page File. The use of SSD volume ensures high speed access, yet the general purpose type means costs are less than provisioned IOPS volumes.◦ A separate, smaller HDD volume for application binaries, patches and updates. This is because these items do not require high speed access.◦ One gp2 or gp3 volume for log files, traces and dumps. Like the swap volume, the general purpose SSD ensures cost-effective but speedy write capability.◦ And finally, one or more separate provisioned IOPS io1 or io2 volumes to host data files. For large databases, this can be even further broken down into two categories:<ul style="list-style-type: none">● Provisioned io1 volumes for static, low-traffic, or lookup tables● Provisioned io2 or io2 block express volumes for large, transactional tables.● Another way to make this division is to allow a separate volume for transaction logs, redo logs or write-ahead logs, and a separate volume for actual data files. <p>Separation of volumes for different components not only gives the flexibility to “mix-and-match” for optimal performance and cost, it also ensures the volumes persist even if the EC2 instance crashes, or if there's a need to forensically analyze the volumes. It also gives the ability to encrypt individual volumes.</p>
Availability	<ul style="list-style-type: none">◦ Configure EBS snapshot for all application and data volumes across all EC2 instances.◦ Use synchronized EBS snapshots for all volumes related to the same application, platform, or process workflow. For EBS volumes attached to the same EC2 instance, this can be achieved with EBS multi-volume snapshots.◦ Devise an automated snapshot deletion process in line with your data retention policy. You can use Amazon Data Lifecycle Manager for this.◦ Implement automated, periodic restore and mounting of EBS volumes from snapshots and check their data consistency.

Area	Recommended Best Practices
Security	<ul style="list-style-type: none">◦ Enable encryption for existing EBS volumes with AWS-managed or customer-managed KMS keys.◦ Ensure EBS volumes are configured for encryption by default.◦ Implement automatic EBS encryption key rotation.
Monitoring	<ul style="list-style-type: none">◦ Implement automated alerting for the failed EBS snapshot jobs.◦ Use automated monitoring and alerting for impaired EBS volumes.◦ Create EBS performance monitoring dashboards in CloudWatch that can show<ul style="list-style-type: none">● Volumes experiencing throttling.● Consistently under-utilized volumes.
Performance	<ul style="list-style-type: none">◦ Use EBS-optimized EC2 instances (or EC2 instances with 10 GB networking) for high-performance applications.◦ Use current generation EBS-optimized EC2 instance types.◦ Ensure the throughput capacity of the EC2 instance closely matches the throughput capacity of its attached EBS volumes.◦ Choose an appropriate block size for the intended workload when formatting the volume with a file system. For example, Oracle databases have a default block size of 8 KB. If your EBS volume hosts Oracle DBF files, it's worthwhile to use an 8 KB block size.◦ Ensure the EBS snapshot restore process utilizes FSR.
Cost	<ul style="list-style-type: none">◦ Create an automated process to identify and delete detached EBS volumes.◦ Ensure EC2 instances have the DeleteOnTermination property set to True for all attached EBS volumes.◦ Use a consistent tagging mechanism for all EBS volumes and snapshots.◦ Identify and remove already-attached EBS volumes that are not hosting any data, application, or swap space.◦ For SSD volumes, use General Purpose instead of Provisioned IOPS unless application performance requires otherwise.◦ Consider using HDD volumes for small-scale applications where high performance isn't important (e.g., small look-up databases, firewalls, etc.).

Conclusion

This e-book provides a comprehensive examination of AWS EBS, specifying its various features, benefits, and drawbacks. We outlined the available file systems and volume types as well as their impact on speed, throughput and availability and went through some strategies for analyzing and optimizing storage volumes. We described tips for cost efficient EBS utilization to ensure users avoid overspending on EBS.

Finally, we discussed common practices you should avoid when using EBS and some best practices to ensure you get the most out of this popular block storage service.

Hopefully, this ebook has provided enough information for you to have a closer look at your EBS volumes, save costs, and fine-tune performance. Remember, Amazon is always changing its disk prices and features. To keep up with the changes, refer back to their documentation.