

Cloud Transformation

Data Management

CraigKeenan.Info: Lesson and Lab Guide

Copyright 2017 craigkeenan.info, and/or its affiliates. All rights reserved.

CraigKeenan.info trademarks and trade dress may not be used in conjunction with any product or services that is not CraigKeenan.info's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits CraigKeenan.Info. All other trademarks not owned by CraigKeenan.Info are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by CraigKeenan.Info.

Lesson Primer

Data Management and Storage Architecture

Data Management - What and Why?

URL Link <https://www.techopedia.com/definition/5422/data-management>

Data management refers to an organization's management of information and data for secure and structured access and storage.

Data management tasks include the creation of data governance policies, analysis and architecture; database management system (DMS) integration; data security and data source identification, segregation and storage.

HINT While we have built our environment, we have not created any data. Once we start posting to our website or uploading files, we need to give some thought to protecting our data. Think along the lines of your family photos, or important documents. We want to protect our data until we no longer need, and it can safely be deleted.

Storage Architectures (Block, File and Object)

For those unfamiliar with Storage Management in an IT environment, it is beyond this course to go into a technical review. However, we will try and provide an overview. Understand this is a very large topic, and very important to most corporations. Network and Storage costs are the largest strain on most IT budgets.

At a very high level there are three types of storage system:

Block Level Storage – is raw storage volumes that are presented to a server as a disk. The server then formats the disk creating a volume and a file system which is presented as a mount or drive. Think of a hard disk in your PC, any Direct Attached Storage(DAS), or even a thumb drive. In the enterprise, there are massive Storage Arrays from vendor like IBM, HP and EMC. These high-end performance arrays are typically connected on a Storage Area Network (SAN), and the servers use a special fiber Host Bus Adapter to connect to the SAN and to the Storage Array. Block Level Storage volumes are typically presented to only one server, but can be attached to multiple servers if the servers support clustered block level storage file locking.

File Level Storage – is storage that is presented to many servers as a file system. Examples of File Level Storage are NFS, SMB (CIFS). This is commonly referred to as Network Attached Storage (NAS) in the business world. However, NAS devices are also commercially available to consumers.

Object Level Storage – is storage for objects, or blobs of data. Each object typically includes the data itself, a variable amount of metadata, and a globally unique identifier. Think S3 Bucket Objects.

Magnetic (or Tape) – Tape or Virtual Tape backups. For the average consumer, you may have purchased an external hard drive and perform backups to disk. In the past in the corporate world, in an effort to manage costs, backups were traditionally written to tape.

WORM – is Write Once, Read Many storage. You won't see much of this in the business world today. However, for archive purpose, some companies used Optical Platters for long term retention. Think CD/DVD-ROMs.

Storage Type Use Cases

- Block Level Storage is the highest performance, most resilient and most expensive (can be millions of \$USD).
- File Level Storage is a lower cost solution and address situations where multiple servers need to share storage.

- Object Level Storage is an ideal lower cost solution for unstructured data (i.e. Photos, Video, Logs, etc)
- Magnetic (or Tape) is for backup / archive.

Amazon Web Services (AWS)

[AWS Storage Services Overview – White Paper](#)

This is an important AWS White Paper. If you intend to take any of the AWS Certified Exams, this is a MUST READ.

We will review most of the material in this lesson, and discuss the remaining information in later lessons.

[AWS Resources Geographic Context](#)

It is important that you understand that AWS resources are either provisioned globally, regionally or in an Availability Zone. You should start to learn the availability of resources (Global, Regional or tied to an Availability Zone). This is important as you design your environment to withstand failures.

[Amazon Web Services Overview](#)

Simple Storage Service (S3)

Amazon Simple Storage Service (Amazon S3) is storage for the Internet. You can use Amazon S3 to store and retrieve any amount of data at any time, from anywhere on the web.

Elastic Compute Cloud (EC2)

Amazon Elastic Compute Cloud (Amazon EC2) is a web service that provides resizable computing capacity—literally, servers in Amazon's data centers—that you use to build and host your software systems.

Elastic Block Store (EBS)

Amazon Elastic Block Store (Amazon EBS) provides persistent block storage volumes for use with Amazon EC2 instances in the AWS Cloud. Each Amazon EBS volume is automatically replicated within its Availability Zone to protect you from component failure, offering high availability and durability. Amazon EBS volumes offer the consistent and low-latency performance needed to run your workloads.

Elastic File System (EFS)

Amazon EFS is a shared file storage service for Amazon EC2 instances. Amazon EFS is easy to use and provides a simple interface that allows you to create and configure file systems quickly and easily. With Amazon EFS, storage capacity is elastic, growing and shrinking automatically as you add and remove files, so your applications have the storage they need, when they need it.

Relational Database Service (RDS)

Amazon Relational Database Service (Amazon RDS) is a web service that makes it easier to set up, operate, and scale a relational database in the cloud. It provides cost-efficient, resizable capacity for an industry-standard relational database and manages common database administration tasks.

Simple Storage Service (S3)

[Amazon S3 Storage Classes](#)

Amazon S3 offers a range of storage classes designed for different use cases. These include Amazon S3 Standard for general-purpose storage of frequently accessed data, Amazon S3 Standard - Infrequent Access for long-lived, but less frequently accessed data, and Amazon Glacier for long-term archive. Amazon S3 also offers configurable lifecycle policies for managing your data throughout its lifecycle.

[Amazon S3 Using Versioning](#)

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures.

[Amazon S3 Object Lifecycle Management](#)

Lifecycle configuration enables you to specify the lifecycle management of objects in a bucket. The configuration is a set of one or more rules, where each rule defines an action for Amazon S3 to apply to a group of objects. For example, you may choose to transition objects to the STANDARD_IA (IA, for infrequent access) storage class 30 days after creation, or archive objects to the GLACIER storage class one year after creation.

[Amazon S3 Object Tagging](#)

Object tagging enables you to categorize storage. Each tag is a key-value pair.

[Cross-Region Replication](#)

Cross-region replication is a bucket-level feature that enables automatic, asynchronous copying of objects across buckets in different AWS regions. To activate this feature, you add a `replication` configuration to your source bucket. In the configuration, you provide information such as the destination bucket where you want objects replicated to. You can request Amazon S3 to replicate all or a subset of objects with specific key name prefixes.

The object replicas in the destination bucket are exact replicas of the objects in the source bucket. They have the same key names and the same metadata

[Amazon S3 Transfer Acceleration](#)

Amazon S3 Transfer Acceleration enables fast, easy, and secure transfers of files over long distances between your client and an S3 bucket. Transfer Acceleration takes advantage of Amazon CloudFront's globally distributed edge locations.

[Amazon Glacier](#)

[What Is Amazon Glacier?](#)

Amazon Glacier is an extremely low-cost storage service that provides durable storage with security features for data archiving and backup. With Amazon Glacier, customers can store their data cost effectively for months, years, or even decades.

Amazon Glacier is a great storage choice when low storage cost is paramount, your data is rarely retrieved, and retrieval latency of several hours is acceptable.

Important Amazon Glacier provides a management console, which you can use to create and delete vaults. However, all other interactions with Amazon Glacier require that you use the AWS Command Line Interface (CLI) or write code.

HINT Glacier requires you to use the CLI or write code. We WON'T be doing labs for Glacier in this lesson.

Elastic Compute Cloud (EC2)

[Amazon AMI](#)

An **Amazon Machine Image (AMI)** provides the information required to launch an instance, which is a virtual server in the cloud. You specify an AMI when you launch an instance, and you can launch as many instances from the AMI as you need. You can also launch instances from as many different AMIs as you need.

[Amazon EC2 Root Device Volume](#)

When you launch an instance, the **root device volume** contains the image used to boot the instance.

When we introduced Amazon EC2, all AMIs were backed by Amazon EC2 instance store, which means the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3. After we introduced Amazon EBS, we introduced AMIs that are backed by Amazon EBS. This means that the root device for an instance launched from the AMI is an Amazon EBS volume created from an Amazon EBS snapshot.

You can choose between AMIs backed by Amazon EC2 instance store and AMIs backed by Amazon EBS. We recommend that you use AMIs backed by Amazon EBS, because they launch faster and use persistent storage.

Elastic Block Storage (EBS)

[Amazon EBS Volumes](#)

An **Amazon EBS** volume is a durable, block-level storage device that you can attach to a single EC2 instance. EBS volumes persist independently from the running life of an EC2 instance. After a volume is attached to an instance, you can use it like any other physical hard drive.

[Creating an Amazon EBS-Backed Linux AMI](#)

To create an Amazon EBS-backed Linux AMI, start from an instance that you've launched from an existing Amazon EBS-backed Linux AMI. After you've customized the instance to suit your needs, create and register a new AMI, which you can use to launch new instances with these customizations.

[EBS Snapshot](#)

After writing data to an EBS volume, you can periodically create a snapshot of the volume to use as a baseline for new volumes or for data backup. If you make periodic snapshots of a volume, the snapshots are incremental so that only the blocks on the device that have changed after your last snapshot are saved in the new snapshot.

[Restoring an Amazon EBS Volume from a Snapshot](#)

You can restore an Amazon EBS volume with data from a snapshot stored in Amazon S3. You need to know the ID of the snapshot you wish to restore your volume from and you need to have access permissions for the snapshot.

[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. For greater I/O performance than you can achieve with a single volume, RAID 0 can stripe multiple volumes together; for on-instance redundancy, RAID 1 can mirror two volumes together.

Amazon EC2 Instance Store

What is Amazon EC2 Instance Store

An **instance store** provides temporary block-level storage for your instance. This storage is located on disks that are physically attached to the host computer. Instance store is ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances.

Add Instance Store Volumes to Your EC2 Instance

You specify the EBS volumes and instance store volumes for your instance using a block device mapping. Each entry in a block device mapping includes a device name and the volume that it maps to.

A block device mapping always specifies the root volume for the instance. The root volume is either an Amazon EBS volume or an instance store volume.

You can specify the instance store volumes for your instance only when you launch an instance. You can't attach instance store volumes to an instance after you've launched it.

Elastic File System

What Is Amazon Elastic File System?

Amazon Elastic File System (Amazon EFS) provides simple, scalable file storage for use with Amazon EC2. With Amazon EFS, storage capacity is elastic, growing and shrinking automatically as you add and remove files, so your applications have the storage they need, when they need it.

Amazon EFS: How It Works

Amazon EFS provides file storage in the AWS Cloud. With Amazon EFS, you can create a file system, mount the file system on an Amazon EC2 instance, and then read and write data from to and from your file system. You can mount an Amazon EFS file system in your VPC, through the Network File System version 4.1 (NFSv4.1) protocol.

You can access your Amazon EFS file system concurrently from Amazon EC2 instances in your Amazon VPC, so applications that scale beyond a single connection can access a file system. Amazon EC2 instances running in multiple Availability Zones within the same region can access the file system, so that many users can access and share a common data source.

To access your Amazon EFS file system in a VPC, you create one or more **mount targets** in the VPC. You mount your file system from these mount targets. A mount target represents an IP address for an NFSv4.1 endpoint at which you can mount an Amazon EFS file system.

Mount targets themselves are designed to be highly available. When designing your application for high availability and the ability to failover to other Availability Zones, keep in mind that the IP addresses and DNS for your mount targets in each Availability Zone are static.

After mounting the file system via the mount target, you use it like any other POSIX-compliant file system. For information about NFS-level permissions and related considerations, see [Network File System \(NFS\)–Level Users, Groups, and Permissions](#).

Getting Started with Amazon Elastic File System

This Getting Started exercise shows you how to quickly create an Amazon Elastic File System (Amazon EFS) file system, mount it on an Amazon Elastic Compute Cloud (Amazon EC2) instance in your VPC, and test the end-to-end setup.

Relational Database Service (RDS)

Working With Backups

Amazon RDS creates and saves automated backups of your DB instance. Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases.

Amazon RDS creates automated backups of your DB instance during the backup window of your DB instance. Amazon RDS saves the automated backups of your DB instance according to the backup retention period that you specify. If necessary, you can recover your database to any point in time during the backup retention period.

Restoring From a DB Snapshot

Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. You can create a DB instance by restoring from this DB snapshot. When you restore the DB instance, you provide the name of the DB snapshot to restore from, and then provide a name for the new DB instance that is created from the restore.

Restoring a DB Instance to a Specified Time

The Amazon RDS automated backup feature automatically creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. Automated backups are kept for a configurable number of days (called the backup retention period). You can restore your DB instance to any specific time during this retention period, creating a new DB instance.

Copying a DB Snapshot or DB Cluster Snapshot

With Amazon Relational Database Service (Amazon RDS), you can copy DB .

As an alternative to copying, you can also share manual snapshots with other AWS accounts.

You can copy snapshots shared to you by other AWS accounts.

WordPress Web Server

Configure http for URL Re-Write

In simple terms, we can store content, such as images, in a different location than on the Web Server, and users browsing our website will be redirected. We will use in our lab to re-direct traffic to CloudFront CDN for image content.

Before you begin

Before you begin your Cloud Transformation, you will need to have think about a few things.

Lesson Perquisites

You must have completed all previous lesson(s) before starting on this course.

Understand that Amazon Free Tier Pricing, so **don't leave services running unnecessarily.**

You only Pay for What You Use

With Amazon Web Services, you only pay for what you use. Since we will be using Free Tier eligible services for most parts of the lecture, our costs should be minimal. Each lesson is design to have you create, modify, and then delete any resources we create. As a matter of best practice, **delete any resources you have created when you are done with the lesson.** If you intend to pause, and then resume hours or even days later... STOP! Go back and **delete any resources you created during this lab to avoid unnecessary charges.**

Note:

We will eventually build out a 'Production' like environment with the intention of running our Website.

When necessary, we will identify through the procedures what resources to leave running. If not specifically stated to leave the resource, it should be deleted upon completion of the lesson.

Understand Data Management Concepts

In simple terms, you need to understand your data needs, identify appropriate storage to store data, protect data from both intentional and malicious corruption, and eventually archive/delete your data.

Know AWS Storage Services

Be familiar with AWS Storage Services (Simple Storage Service (S3), Glacier, EBS, Instance Store, Elastic File System (EFS)). Know the use cases and when not to use.

Understand Durability and Availability

You should understand that not all Storage services offer the same level of data protection, performance or availability across Availability Zones and Regions. Durability is how well protected the data. Availability is how resilient is access to my data.

Practical versus Theory

The intent of these lessons is to provide hands-on labs, and we do. We will provide hundreds of individual procedures. At the same time, there are limits to what you and I can test as individuals with limited resources and budget. We attempt to limit the theoretical conversations at this time, but on some occasions must introduce key concepts to provide a comprehensive experience. For example, we will not Lab Glacier in this lesson. It requires programmatic experience we are not ready to provide you yet. **HINT HINT**

Lab Overview

Throughout all lessons, we will be acting as if we are an individual who is looking to get started with Amazon Web Services Cloud Computing. Fortunately, we have a friend who is a Cloud Guru, and this friend has agreed to guide us through the process. Our goal is to build a web site for our product/service. It can be anything we want it to be. For teaching purposes, we'll call our experiment OurAmazonWebServices (**OAWS**).

Note: When you see an example using **oaws** or **ouraws** in the name, **you must replace** with a name unique to your environment.

We have a WordPress Web Server in a public subnet supported by an RDS DB Instance in a private subnet deployed. Our Cloud friend has suggested we start giving some thought to our data before we go further. At the moment it's no big deal if we lost our site. But once we start posting and uploading content we have to secure our data. We are going to explore several of the AWS Storage Services in this lab. By the end of the lab

we will have protected our data and learned about some new services (some of which we will not be using). Our friend insists we have a comprehensive education.

Again, we are working with a minimal budget so everything we do at this time will be in the Free Tier. Pay attention and make sure you select the correct Free Tier services.

Data Management - Lab Overview

Simple Storage Services (S3)

- ☐ Editing the Details of Objects with a Storage Class of STANDARD, STANDARD_IA or RSS
- ☐ Enabling Bucket Versioning
- ☐ Deleting Objects from a Versioning-Enabled Bucket
- ☐ Add a Lifecycle Configuration Rule to a Bucket without Versioning.
- ☐ Add a Lifecycle Configuration Rule to a Bucket with Versioning.
- ☐ Maintaining Lifecycle Configuration Rules
- ☐ How Do I Add Tags to an Object
- ☐ Enabling Amazon S3 Transfer Acceleration

Elastic Compute Cloud (EC2)

- ☐ Choose an AMI by Root Device Type
- ☐ Creating an Amazon EBS Snapshot
- ☐ Creating a Linux AMI from an Instance
- ☐ Creating a Linux AMI from a Snapshot

Elastic Block Storage (EBS)

- ☐ Creating and Amazon EBS Volume
- ☐ Attaching an EBS Volume to an Instance
- ☐ Making an Amazon EBS Volume Available for Use on Linux
- ☐ Making an Amazon EBS Volume Available for Use on Windows
- ☐ Windows Server – Overview of Disk Management
- ☐ Windows Server – Manage Disks
- ☐ Windows Server – Initialize New Disks
- ☐ Windows Server – Create a Simple Volume
- ☐ Create Amazon EBS Snapshot
- ☐ Restoring an EBS Volume from a Snapshot
- ☐ RAID Configuration on Linux
- ☐ RAID Configuration on Windows
- ☐ Windows Server – Create a Striped Volume
- ☐ Windows Server – Delete a Volume
- ☐ Detaching an Amazon EBS Volume
- ☐ Deleting an Amazon EBS Volume

Amazon EC2 Instance Store

- ☐ Launch Instance Store Backed EC2 Instance in a VPC
- ☐ Stop Your Instance Store Backed EC2 Instance
- ☐ Adding Instance Store Volumes to an Instance
- ☐ Stop Your EBS Backed EC2 Instance with Attached Instance Store Volume

Elastic File System (EFS)

- ☐ Create Your Amazon EFS File System
- ☐ Connect to Your Amazon EC2 Instance and Mount the File System
- ☐ Troubleshooting Amazon EFS
- ☐ Updating DNS Support for Your VPC
- ☐ Deleting an Amazon EFS File System

Relational Database Service (RDS)

- ☐ Enabling Automated Backups
- ☐ Disabling Automated Backups
- ☐ Restoring a DB Instance to a Specified Time
- ☐ Creating a DB Snapshot
- ☐ Restoring from a DB Snapshot
- ☐ Copying a DB Snapshot by Using the AWS Management Console

WordPress Server

- ☐ Uploading Files
- ☐ Copy WordPress Code and Content to S3 Bucket
- ☐ Configure http for URL Re-write
- ☐

Cleanup!

- ☐ Delete / Stop All Billable Resources
- ☐ Environment Configuration

Lab – Data Management

Simple Storage Services (S3)

Editing the Details of Objects with a Storage Class of STANDARD, STANDARD_IA, or RRS

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/EditingObjectDetails.html>

This section describes how to change the property details of an object with a storage class of STANDARD, STANDARD_IA (IA, for infrequent access), or RRS (Reduced Redundancy Storage).

To change the details of an object with a Storage Class of STANDARD, STANDARD_IA, or RRS

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. **Choose the object** you want to change the details for.
3. (Optional) To change the storage class, **select the class** you want to use.
4. Choose **Save** to save your changes.

Enabling Bucket Versioning

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/enable-bucket-versioning.html>

This section describes how to enable versioning on a bucket.

To enable versioning on a bucket

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, **click the details icon** on the left of the bucket name and then **click Properties** to display bucket properties.
3. In the **Properties** pane, **click Versioning** and then **click Enable Versioning**.
4. The console displays a confirmation dialog. **Click OK** to enable versioning on the bucket.

Amazon S3 enables versioning on the bucket. Accordingly, the console UI replaces the **Enable Versioning** button with the **Suspend Versioning** button.

After you enable versioning on a bucket, it can be in only the enabled or suspended state; you cannot disable versioning on a bucket. If you suspend versioning, Amazon S3 suspends the creation of object versions for all operations, but preserves any existing object versions.

HINT Versioning only applies to objects after it has been enabled. Re-upload any existing objects to enable versioning on those objects.

Deleting Objects from a Versioning-Enabled Bucket

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/deleting-object-versioned-bucket.html>

In a versioning-enabled bucket, you can either delete an object from the object list (version information hidden) or delete a specific version of the object.

If you **select** and **delete an object**, Amazon S3 adds a delete marker for the object and the object no longer appears in the object list.

However, if you **click Show** to list object versions, the object appears in the list with all versions and a delete marker at the top.

To delete an object permanently, you must delete all the versions of the object, including the delete marker (if present). If you delete only a specific object version, Amazon S3 permanently deletes only that specific version.

If you delete the delete marker, the object reappears in the object list.

Important Versioning is applied to objects that have been uploaded after versioning has been enabled. Any existing files will have only have current version.

Add a Lifecycle Configuration Rule to a Bucket without Versioning

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/lifecycle-configuration-bucket-no-versioning.html>

An unversioned bucket maintains only one version of each object, and the lifecycle transition and expiration actions apply to these objects.

The following example walkthrough creates a lifecycle configuration rule for a bucket that archives video files in the bucket 90 days after creation and then permanently deletes them 455 days after creation. The rule also automatically ends and cleans up any multipart uploads that have not completed after 7 days.

Add a Lifecycle Configuration Rule to a Bucket without Versioning

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, **choose the bucket** whose lifecycle configuration you want to configure, **click Properties** and then **choose Lifecycle**.
3. Choose **Add rule**.
4. **Select A Prefix** and enter **videos/** as the prefix to specify the subset of objects to which the rule applies, and then **click Configure Rule**. (In our example, entering "videos/" will apply the rule to all objects in the bucket's "videos" folder.)

If you selected **Whole Bucket** the rule would apply to all objects in the bucket.

5. You configure lifecycle rules by defining actions. In the **Action on Objects** section define the lifecycle actions:
 - a. **Select Archive to the Glacier Storage Class** and **enter 90** for the number of days after an object's creation date that you want to archive the object to the Glacier storage class.
Select Permanently Delete and **enter 455** for the number of days after an object's creation date that you want the object to be permanently deleted. You cannot recover permanently deleted objects.
Important Selecting Permanently Delete will not remove incomplete multipart uploads. You must select End and Clean up Incomplete Multipart Uploads as described in the next step to have incomplete multipart uploads removed.
 - b. It is a recommended best practice to select **End and Clean up Incomplete Multipart Uploads**. For our example, **enter 7** for the number of days after the multipart upload initiation date that you want to end and clean up any multipart uploads that have not completed. Then **choose Review**.
6. Review and name your rule.
 - a. (Optional) You can give your rule a name to identify the rule, if you want. The name must be unique within the bucket. By default, Amazon S3 will generate a unique identifier for the rule.
 - b. **Choose Edit** next to **Rule Target** or **Rule Configuration** if you want to make changes.
 - c. **Choose Create and Activate Rule** when all of the settings are as you want them.
7. If the rule does not contain any errors, it is displayed in the **Lifecycle** pane.

Add a Lifecycle Configuration Rule to a Bucket with Versioning

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/lifecycle-configuration-bucket-with-versioning.html>

A versioning-enabled bucket can have many versions of the same object, one current version and zero or more noncurrent (previous) versions. You can add lifecycle rules to buckets that have object versioning enabled or suspended. Using a lifecycle configuration, you can define actions specific to current and noncurrent object versions.

The combined lifecycle and versioning functionality acts like a recycling bin, granting you the following benefits:

- Recovering previous versions for a specified time to protect against unintended overwrites or deletions of your content.
- Setting specific windows of time for retaining the noncurrent versions in Amazon S3, archiving in Amazon Glacier, or scheduling automatic deletion to help you control storage costs.

The following example walkthrough adds the following lifecycle configuration to a versioning-enabled bucket.

- Archive the current object versions in the `documents` folder 365 days after creation.
- Transition noncurrent objects to the `STANDARD_IA` (infrequent access) storage class 30 days after they become noncurrent and transition them to the `GLACIER` storage class (archive them) 60 days after they become noncurrent. Permanently delete the noncurrent objects 425 days after they become noncurrent and remove expired object delete markers.
- End and clean up multipart uploads that have not completed after 7 days.

Add a Lifecycle Configuration Rule to a Versioned Bucket

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, **choose the bucket** whose lifecycle configuration you want to configure, **choose Properties**, and then **choose Lifecycle**.
3. **Choose Add rule**.
4. **Select A Prefix** and **enter documents/** as the prefix to specify the subset of objects to which the rule applies, and then **click Configure Rule**. (In our example, entering "documents/" will apply the rule to all objects in the bucket's "documents" folder.)

If you selected **Whole Bucket** the rule would apply to all objects in the bucket.

5. Configure the rule describing actions for both current and noncurrent (previous) object versions.
 - a. In the **Action on Current Version** section **select the Archive to the Glacier Storage Class** and **specify 365 days**.
 - b. Actions selected in the **Action on Previous Versions** section occur according to the specified number of days after the object becomes noncurrent.
Select Transition to the Standard-Infrequent access Storage Class and **enter 30 days**, and then **select Archive to the Glacier Storage Class** and **enter 60 days**.
Select Permanently Delete and **enter 425 days** and then **select Remove expired object delete marker**. Amazon S3 will remove an expired object delete marker no sooner than 48 hours after the object expired.

Important Selecting **Permanently Delete** will not remove incomplete multipart uploads. You must select **End and Clean up Incomplete Multipart Uploads** as described in the next step to have incomplete multipart uploads removed.

- c. It is a recommended best practice to **select End and Clean up Incomplete Multipart Uploads**. For our example, **enter 7** for the number of days after the multipart upload initiation date that you want to end and clean up any multipart uploads that have not completed. Then **choose Review**.
6. Review and name your rule.
 - a. (Optional) You can give your rule a name to identify the rule, if you want. The name must be unique within the bucket. By default, Amazon S3 will generate a unique identifier for the rule.
 - b. **Choose Edit** next to **Rule Target** or **Rule Configuration** if you want to make changes.
 - c. **Choose Create and Activate Rule** when all of the settings are as you want them.
7. If the rule does not contain any errors, it is displayed in the **Lifecycle** pane.

Maintaining Lifecycle Configuration Rules

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/UG/lifecycle-configuration-bucket-maintain-rules.html>

The **Lifecycle** pane of the bucket **Properties** show the lifecycle rules that you have configured on the bucket.

You can **edit a rule or delete a rule**. Also, you can disable a rule by clearing the check box next to the rule. When a rule is disabled, Amazon S3 does not perform any actions defined in the rule.

If you have configured a lifecycle rule on a bucket to expire objects in that bucket, each object that the rule applies will have its object properties display the date when the object will expire and the lifecycle rule that has set the expiration action on the object, as shown in the following screen shot.

How Do I Add Tags to an Object

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/user-guide/add-object-tags.html>

This section explains how to use the console to add tags to an object.

To add tags to an object

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. In the **Bucket name** list, **choose the name of the bucket** that contains the object.
3. In the **Name** list, **choose the name of the object** you want to add tags to.
4. **Choose Properties**.
5. **Choose Tags** and then **choose Add Tag**.
6. Each tag is a key-value pair. **Type a Key and a Value**. Then choose **Add Tag** to add another tag or **choose Save**.

You can enter up to 10 tags for an object.

Walkthrough 1: Configure Cross-Region Replication Where Source and Destination Buckets Are Owned by the Same AWS Account

URL Link <http://docs.aws.amazon.com/AmazonS3/latest/dev/crr-walkthrough1.html>

In this section, you create two buckets (source and destination) in different AWS regions, enable versioning on both the buckets, and then configure cross-region replication on the source bucket.

1. **Create two buckets**.
 - a. **Create a source bucket in an AWS region**. For example, US West (Oregon) (us-west-2). For instructions, see [Creating a Bucket](#) in the Amazon Simple Storage Service Console User Guide.
 - b. **Create a destination bucket in another AWS region**. For example, US East (N. Virginia) region (us-east-1).

2. **Enable versioning on both buckets.** For instructions, see [Enabling Bucket Versioning](#) in the Amazon Simple Storage Service Console User Guide.
3. **Enable cross-region replication on the source bucket.** You decide if you want to replicate all objects or only objects with a specific prefix (when using the console, think of this as deciding if you want to replicate only objects from a specific folder). For instructions, see [Enabling Cross-Region Replication](#) in the Amazon Simple Storage Service Console User Guide.
4. **Test the setup as follows:**
 - a. Create objects in the source bucket and verify that Amazon S3 replicated the objects in the destination bucket. The amount of time it takes for Amazon S3 to replicate an object depends on the object size. For information about finding replication status, see [How to Find Replication Status of an Object](#).

Remember that the replicas are exact copies of the objects in the source bucket.

Enabling Amazon S3 Transfer Acceleration

URL Link <http://docs.aws.amazon.com/AmazonS3/latest/UG/enable-bucket-transfer-acceleration.html>

This section describes how to enable Amazon S3 Transfer Acceleration on a bucket. For more information about transfer acceleration in Amazon S3, see [Transfer Acceleration](#) in the Amazon Simple Storage Service Developer Guide.

To enable Transfer Acceleration on a bucket

1. Sign in to the AWS Management Console and open the **Amazon S3 console** at <https://console.aws.amazon.com/s3/>.
2. In the **Buckets** list, **choose the bucket** you want to enable, **choose Properties**, and then **choose Transfer Acceleration**.
3. **Choose Enable** to enable Transfer Acceleration.
4. Amazon S3 enables Transfer Acceleration on the bucket. Accordingly, the **Enable** button text changes to **Suspend**.

Endpoint displays the endpoint domain name that you use to access accelerated data transfers to and from the Transfer Acceleration enabled bucket. If you suspend Transfer Acceleration, the accelerate endpoint will no longer be displayed and will no longer work.

Note Endpoint

5. (Optional) Choose **Want to compare your data transfer speed by region?** if you want to run the Amazon S3 Transfer Acceleration Speed Comparison tool, which compares accelerated and non-accelerated upload speeds starting with the region of the enabled bucket. The Speed Comparison tool uses multipart uploads to transfer a file from your browser to various Amazon S3 regions with and without using Amazon S3 Transfer Acceleration.

HINT Suspend S3 Transfer Acceleration when complete

Elastic Compute Cloud (EC2)

Choosing an AMI by Root Device Type

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/RootDeviceStorage.html>

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

To choose an Amazon EBS-backed AMI using the console

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

To choose an instance store-backed AMI using the console

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

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

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

Creating an Amazon EBS Snapshot

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html>

To create a snapshot using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. Choose **Snapshots** in the navigation pane.
3. Choose **Create Snapshot**.
4. In the **Create Snapshot** dialog box, select the volume to create a snapshot for, and then choose **Create**.

(Ex.

Name = oawsSnap

Description = OAWS EBS Snapshot

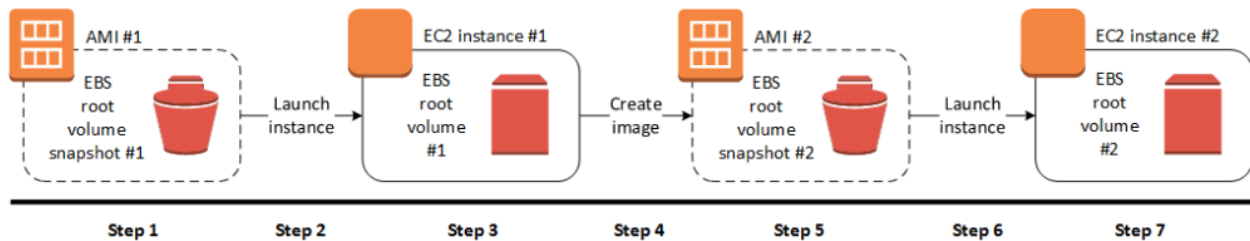
)

Hint If you have more than one EC2 Instance, it can be confusing which volume to snapshot. Review **Volumes**, Select an available volume, in **Description** tab of lower pane, record **Attachment information**. This is your **Instance ID** and **Tag**.

Creating a Linux AMI from an Instance

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html>

You can create an AMI using the AWS Management Console or the command line. The following diagram summarizes the process for creating an Amazon EBS-backed AMI from a running EC2 instance. Start with an existing AMI, launch an instance, customize it, create a new AMI from it, and finally launch an instance of your new AMI.



To create an AMI from an instance using the console

1. Select an appropriate EBS-backed AMI to serve as a starting point for your new AMI, and configure it as needed prior to launch.
2. Choose Launch to launch an instance of the EBS-backed AMI that you've selected. Accept the default values as you step through the wizard.
3. While the instance is running, connect to it.

You can perform any of the following actions on your instance to customize it for your needs:

- Install software and applications
- Copy data
- Reduce start time by deleting temporary files, defragmenting your hard drive, and zeroing out free space
- Attach additional Amazon EBS volumes

(Optional) Create snapshots of all the volumes attached to your instance.

In the navigation pane, choose Instances and select your instance. Choose Actions, Image, and Create Image.

Tip If this option is disabled, your instance isn't an Amazon EBS-backed instance.

4. In the Create Image dialog box, specify values for the following fields, and then choose Create Image.

Image Name A unique name for the image.

Image Description (Optional) A description of the image, up to 255 characters.

(Ex.

Name = oawsAMI

Description = OAWS AMI

)

By default, Amazon EC2 shuts down the instance, takes snapshots of any attached volumes, creates and registers the AMI, and then reboots the instance. Choose **No reboot** if you don't want your instance to be shut down.

Warning If you choose **No reboot**, we can't guarantee the file system integrity of the created image.

5. While your AMI is being created, you can choose AMIs in the navigation pane to view its status. Initially this will be pending. After a few minutes the status should change to *available*.
(Optional) Choose Snapshots in the navigation pane to view the snapshot that was created for the new AMI. When you launch an instance from this AMI, we use this snapshot to create its root device volume.
6. Launch an instance from your new AMI.
7. The new running instance contains all of the customizations you applied in previous steps.

Creating a Linux AMI from a Snapshot

If you have a snapshot of the root device volume of an instance, you can create an AMI from this snapshot using the AWS Management Console or the command line.

To create an AMI from a snapshot using the console

1. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.

2. In the navigation pane, under **Elastic Block Store**, choose **Snapshots**.
3. Choose the snapshot and choose **Actions, Create Image**.
4. In the **Create Image from EBS Snapshot** dialog box, complete the fields to create your AMI, then choose **Create**.

Name A unique name for the image.

Description (Optional) A description of the image, up to 255 characters.

(Ex.

Name = oawsSnapAMI

Description = OAWS SNAP AMI

)

Elastic Block Storage (EBS)

Creating an Amazon EBS Volume

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-volume.html>

You can create an Amazon EBS volume that you can then attach to any EC2 instance within the same Availability Zone.

To create an EBS volume using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. From the navigation bar, select the region in which you would like to create your volume. This choice is important because some Amazon EC2 resources can be shared between regions, while others can't.
3. In the navigation pane, under **ELASTIC BLOCK STORE**, choose **Volumes**.
4. Above the upper pane, choose **Create Volume**.
5. In the **Create Volume** dialog box, for **Volume Type**, choose **General Purpose SSD (GP2)**, **Provisioned IOPS SSD (IO1)**, **Throughput Optimized HDD (ST1)**, **Cold HDD (SC1)**, or **Magnetic**.
6. For **Size**, enter the size of the volume, in GiB.
(Ex. 5)
7. For io1 volumes, in the **IOPS** field, enter the maximum number of input/output operations per second (IOPS) that the volume should support.
8. For **Availability Zone**, select the Availability Zone in which to create the volume.
9. Choose **Yes, Create**.

Attaching an Amazon EBS Volume to an Instance

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-attaching-volume.html>

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

Prerequisites

Determine the device names that you'll use.

Determine how many volumes you can attach to your instance.

If a volume is encrypted, it can only be attached to an instance that supports Amazon EBS encryption.

To attach an EBS volume to an instance using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Volumes**.
3. Select a volume and choose **Actions, Attach Volume**.

4. In the **Attach Volume** dialog box, start **typing the name or ID** of the instance to attach the volume to for **Instance**, and **select it** from the list of suggestion options (only instances that are in the same Availability Zone as the volume are displayed).
5. You can keep the suggested device name, or enter a different supported device name.
Important The block device driver for the instance assigns the actual volume name when mounting the volume, and the name assigned can be different from the name that Amazon EC2 recommends.
6. **Choose Attach.**
7. Connect to your instance and make the volume available.

Making an Amazon EBS Volume Available for Use on Linux

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-using-volumes.html>

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.

Note that you can take snapshots of your EBS volume for backup purposes or to use as a baseline when you create another volume.

To make an EBS volume available for use on Linux

1. **Connect to your instance using SSH.**
2. 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.

```
[ec2-user ~]$ lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
xvdf  202:80  0  100G  0 disk
xvda1 202:1   0    8G  0 disk /
```

The output of `lsblk` removes the `/dev/` prefix from full device paths. In this example, `/dev/xvda1` is mounted as the root device (note the MOUNTPOINT is listed as `/`, the root of the Linux file system hierarchy), and `/dev/xvdf` is attached, but it has not been mounted yet.

3. Determine whether you need to create a file system on the volume. New volumes are raw block devices, and you need to create a file system on them before you can mount and use them. Volumes that have been restored 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 the `sudo file -s` device command to list special information, such as file system type.

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

If the output of the previous command shows simply *data* for the device, then there is no file system on the device and you need to create one. You can go on to Step 4.

```
/dev/xvdf: data
```

If you run this command on a device that contains a file system, then your output will be different.

```
/dev/xvda1: Linux rev 1.0 ext4 filesystem data, UUID=1701d228-
e1bd-4094-a14c-8c64d6819362 (needs journal recovery) (extents)
(large files) (huge files)
```

In the previous example, the device contains Linux rev 1.0 ext4 filesystem data, so this volume does not need a file system created (you can skip Step 4 if your output shows file system data).

4. (Conditional) Use the following command to create an ext4 file system on the volume. Substitute the device name (such as /dev/xvdf) for device_name. Depending on the requirements of your application or the limitations of your operating system, you can choose a different file system type, such as ext3 or XFS.
Caution This step assumes that you're mounting an empty volume. If you're mounting a volume that already has data on it (for example, a volume that was restored from a snapshot), don't use **mkfs** before mounting the volume (skip to the next step instead). Otherwise, you'll format the volume and delete the existing data.

```
[ec2-user ~]$ sudo mkfs -t ext4 device_name
```
5. Use the following 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. Substitute a location for mount_point, such as /data.

```
[ec2-user ~]$ sudo mkdir mount_point
```
6. Use the following command to mount the volume at the location you just created.

```
[ec2-user ~]$ sudo mount device_name mount_point
```
7. Review the file permissions of your new volume mount to make sure that your users and applications can write to the volume.

Create a test file in new filesystem

1. Send text to file in filesystem on volume

```
[ec2-user ~]$ sudo echo "Test" > mountpoint/file.test
```

To stop using an EBS volume on Linux

1. Unmount the file system

```
[ec2-user ~]$ sudo umount mountpoint
```

Making an Amazon EBS Volume Available for Use on Windows

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/ebs-using-volumes.html>

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.

Note that you can take snapshots of your EBS volume for backup purposes or to use as a baseline when you create another volume.

To use an EBS volume

2. Log in to your Windows instance using Remote Desktop.
3. Start the **Disk Management utility**. On Windows Server 2012, on the taskbar, open the context (right-click) menu for the Windows logo and choose **Disk Management**. On Windows Server 2003/2008, choose **Start, Administrative Tools, Computer Management, and Disk Management**.
4. In the lower pane, select the disk that represents the new EBS volume.
5. On the **Disk Management** menu, choose **Action, All Tasks, Online**.
(Conditional) A new disk needs to be initialized before it can be used. **Caution** 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), make sure that you don't reformat the volume and delete the existing data.
To initialize a new disk:
 - a. In the Disk Management utility, select the new EBS volume disk.
 - b. On the **Disk Management** menu, choose **Action, All Tasks, Initialize Disk**.

- c. In the **Initialize Disk** dialog, select the disk to initialize, select the desired partition style, and choose **OK**.

Create a test file in new filesystem

1. Create text file in filesystem on volume.
In File Explorer, select New Volume, (right click) in main window, select New, Text Document
Name file, example file.text

To stop using an EBS volume on Windows

1. In **Disk Management** utility, right-click volume, Choose **Change Drive Letter and Paths...**, Choose **Remove**, Click **Yes** to remove.
2. (Optionally) You can delete the volume. In **Disk Management** utility, right-click volume, Choose **Delete Volume...**, Click **Yes** to delete.

HINT We thought AWS procedures was a bit light on demonstrating Windows Disks, Volumes and Drives. The following sections title 'Windows Server –' are provided for additional reference. If you successfully added the volume to the server, you can ignore the additional material.

Windows Server – Overview of Disk Management

URL Link: [https://technet.microsoft.com/en-us/library/cc754936\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc754936(v=ws.11).aspx)

Disk Management is a system utility for managing hard disks and the volumes or partitions that they contain. With Disk Management, you can initialize disks, create volumes, and format volumes with the FAT, FAT32, or NTFS file systems. Disk Management enables you to perform most disk-related tasks without restarting the system or interrupting users. Most configuration changes take effect immediately.

Windows Server – Manage Disks

URL Link: [https://technet.microsoft.com/en-us/library/cc771607\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc771607(v=ws.11).aspx)

This section includes the most common disk management tasks, including moving disks between computers, changing disks between basic and dynamic types, and changing the partition style of disks.

Converting disk types

Although Disk Management allows you to change disks between various types and partition styles, some of the operations are irreversible (unless you reformat the drive). You should carefully consider the disk type and partition style that is most appropriate for your application.

Online and offline status

Disk Management displays the online and offline status of disks.

If a disk is offline, you must bring it online before you can initialize it or create volumes on it. Bring a disk online or take it offline by right-clicking the disk name and then clicking the appropriate action.

Windows Server – Initialize New Disks

URL Link: [https://technet.microsoft.com/en-us/library/cc771486\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc771486(v=ws.11).aspx)

Applies To: Windows 7, Windows Server 2008 R2

Backup Operator or **Administrator** is the minimum membership required.

To initialize new disks

1. In **Disk Management**, right-click the disk you want to initialize, and then click **Initialize Disk**.
2. In the **Initialize Disk** dialog box, select the disk(s) to initialize. You can select whether to use the master boot record (MBR) or GUID partition table (GPT) partition style.
3. Click **Ok**.

Note

The disk is initialized as a basic disk.

Additional considerations

New disks appear as **Not Initialized**. Before you can use a disk, you must first initialize it. If you start Disk Management after adding a disk, the Initialize Disk Wizard appears so you can initialize the disk.

Windows Server – Create a Simple Volume

URL Link: [https://technet.microsoft.com/en-us/library/cc725610\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc725610(v=ws.11).aspx)

Applies To: Windows 7, Windows Server 2008 R2

A simple volume is a dynamic volume that is made up of disk space from a single dynamic disk. A simple volume can consist of a single region on a disk or multiple regions of the same disk that are linked together. You can create simple volumes only on dynamic disks.

Simple volumes are not fault tolerant.

Backup Operator or **Administrator** is the minimum membership required to complete the actions below.

Creating a simple volume

To create a simple volume using the Windows interface

1. In **Disk Management**, right-click the unallocated space on the dynamic disk on which you want to create the simple volume, and then click **New Simple Volume**.
2. In the New Volume Wizard, click **Next**, click **Simple**, and then follow the instructions on your screen.

To create a simple volume using a command line

Open a command prompt and type diskpart.

At the **DISKPART** prompt, type list disk. Make note of the number of the disk where you want to create a simple volume.

At the **DISKPART** prompt, type create volume simple [size=<size>] [disk=<disknumber>].

At the **DISKPART** prompt, type assign letter=<driveletter>.

Creating an Amazon EBS Snapshot

URL Link <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html>

To create a snapshot using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. Choose **Snapshots** in the navigation pane.
3. Choose **Create Snapshot**.
4. In the Create Snapshot dialog box, select the volume to create a snapshot for, and then choose **Create**.
(Ex.


```
Name = oawsEBSSnap
Description = OAWS EBS Snapshot
)
```

Restoring an Amazon EBS Volume from a Snapshot

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-restoring-volume.html>

Prerequisite

- Create an Amazon EBS Snapshot (of Non-Root Device)

To restore an EBS volume from a snapshot using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. From the navigation bar, **select the region** that your snapshot is located in. This choice is important because some Amazon EC2 resources can be shared between regions, while others can't. If you need to restore the snapshot to a volume in a different region, you can copy your snapshot to the new region and then restore it to a volume in that region.
3. In the navigation pane, **choose Volumes, Create Volume**.
4. In the **Create Volume** dialog box, for **Volume Type**, choose **General Purpose SSD, Provisioned IOPS SSD, or Magnetic**.
5. For **Snapshot**, start **typing the ID or description** of the snapshot from which you are restoring the volume, and **select it** from the list of suggested options.

Note Volumes that are restored from encrypted snapshots can only be attached to instances that support Amazon EBS encryption.

6. For **Size**, **enter the size** of the volume in GiB, or verify that the default size of the snapshot is adequate. If you specify both a volume size and a snapshot ID, the size must be equal to or greater than the snapshot size. When you select a volume type and a snapshot ID, minimum and maximum sizes for the volume are shown next to the **Size** list. Any AWS Marketplace product codes from the snapshot are propagated to the volume.
7. For io1 volumes, in the **IOPS** field, enter the maximum number of input/output operations per second (IOPS) that the volume can support.
8. In the **Availability Zone** list, **select the Availability Zone** in which to create the volume. EBS volumes can only be attached to EC2 instances within the same Availability Zone.
9. **Choose Yes, Create**.

Important If you restored a snapshot to a larger volume than the default for that snapshot, you need to extend the file system on the volume to take advantage of the extra space.

After you've restored a volume from a snapshot, you can **attach it to an instance** to begin using it.

RAID Configuration on Linux

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/raid-config.html>

Use the following procedure to create the RAID array.

To create a RAID array on Linux

1. **Create the Amazon EBS volumes** for your array.
Important 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.
2. **Attach the Amazon EBS volumes to the instance** that you want to host the array.
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.

(RAID 0 only) To create a RAID 0 array, execute 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
```

(RAID 1 only) To create a RAID 1 array, execute the following command (note the `--level=1` option to mirror the array):

```
[ec2-user ~]$ sudo mdadm --create --verbose /dev/md0 --level=1 --
name=MY_RAID --raid-devices=number_of_volumes device_name1
device_name2
```

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
```

This yields output such as the following: Personalities : [raid1]

```
md0 : active raid1 xvdg[1] xvdf[0]
20955008 blocks super 1.2 [2/2] [UU]
[=====>.....] resync = 46.8% (9826112/20955008) finish=2.9min
speed=63016K/sec In general, you can display detailed information about your RAID
array with the following command: [ec2-user ~]$ sudo mdadm --detail /dev/md0 This
yields information such as the following: /dev/md0:
Version : 1.2
Creation Time : Mon Jun 27 11:31:28 2016
Raid Level : raid1
Array Size : 20955008 (19.98 GiB 21.46 GB)
Used Dev Size : 20955008 (19.98 GiB 21.46 GB)
Raid Devices : 2
Total Devices : 2
Persistence : Superblock is persistent

Update Time : Mon Jun 27 11:37:02 2016
State : clean
...
...
...
```

Number	Major	Minor	RaidDevice	State	
0	202	80	0	active sync	/dev/sdf
1	202	96	1	active sync	/dev/sdg

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`, execute 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. Create a mount point for your RAID array.

```
[ec2-user ~]$ sudo mkdir -p /mnt/raid
```

7. 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.

Create a test file in new filesystem

2. Send text to file in filesystem on volume

```
[ec2-user ~]$ sudo echo "Test" > /mnt/raid/file.raid.test
```

To stop using an EBS volume on Linux

6. Unmount the file system

```
[ec2-user ~]$ sudo umount /mnt/raid
```

RAID Configuration on Windows

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/raid-config.html>

Use the following procedure to create the RAID array.

To create a RAID array on Windows

1. Create the Amazon EBS volumes for your array.

Important 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.

2. Attach the Amazon EBS volumes to the instance that you want to host the array.
3. Connect to your Windows instance.
4. Open a command prompt and type the diskpart command.

```
PS C:\Users\Administrator> 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		
Disk 3	Online	8 GB	0 B		
Disk 4	Online	8 GB	0 B		
Disk 5	Online	419 GB	0 B		
Disk 6	Online	419 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
```

```
XENSRV 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.

```
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	*	
Disk 3	Online	8 GB	0 B	*	
* Disk 4	Online	8 GB	0 B	*	
Disk 5	Online	419 GB	0 B		
Disk 6	Online	419 GB	0 B		

8. Create your raid array. On Windows, a RAID 0 volume is referred to as a striped volume and a RAID 1 volume is referred to as a mirrored volume.

(Striped volumes only) 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.

(Mirrored volumes only) To create a mirrored volume array on disks 3 and 4, use the following command (note the **mirror** option to mirror the array):

```
DISKPART> create volume mirror disk=3,4
```

DiskPart successfully created the volume.

9. Verify your new volume.

DISKPART> **list volume**

Volume ###	Ltr	Label	Fs	Type	Size	Status	Info
Volume 0	C		NTFS	Partition	29 GB	Healthy	System
* Volume 1			RAW	Mirror	8190 MB	Healthy	
Volume 2			RAW	Stripe	15 GB	Healthy	
Volume 5	Z	Temporary S	NTFS	Partition	419 GB	Healthy	
Volume 6	Y	Temporary S	NTFS	Partition	419 GB	Healthy	

Note that for this example the *Type* column lists a *Mirror* volume and 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 a test file in new filesystem

2. Create text file in filesystem on RAID volume.

In File Explorer, select New Volume, (right click) in main window, select New, Text Document

Name file, example file.raid.text

To stop using an EBS volume on Windows

3. In **Disk Management** utility, **right-click volume**, **Choose Change Drive Letter and Paths...**, **Choose Remove**, **Click Yes** to remove.
4. (Optionally) You can delete the volume. In **Disk Management** utility, right-click volume, Choose **Delete Volume...**, Click **Yes** to delete.

HINT We thought AWS procedures were overly complex for those not familiar with DISKPART command. The following sections title 'Windows Server –' are provided for additional reference. If you successfully added the RAID Volumes to the server, you can ignore the additional material.

[Windows Server – Create a Striped Volume](#)

URL Link: [https://technet.microsoft.com/en-us/library/cc725610\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc725610(v=ws.11).aspx)

A striped volume is a dynamic volume that stores data in stripes on two or more physical disks. Data in a striped volume is allocated alternately and evenly (in stripes) across the disks. Striped volumes offer the best performance of all the volumes that are available in Windows, but they do not provide fault tolerance. If a disk in a striped volume fails, the data in the entire volume is lost.

You can create striped volumes only on dynamic disks. Striped volumes cannot be extended. You can create a striped volume onto a maximum of 32 dynamic disks.

Backup Operator or **Administrator** is the minimum membership required to complete the actions below.

To create a striped volume using the Windows interface

1. In **Disk Management**, right-click the unallocated space on one of the dynamic disks where you want to create the striped volume, and then click **New Striped Volume...**
2. Follow the instructions on your screen.

Windows Server – Delete a Volume

URL Link: [https://technet.microsoft.com/en-us/library/cc772461\(v=ws.11\).aspx](https://technet.microsoft.com/en-us/library/cc772461(v=ws.11).aspx)

You can delete any volume listed on the Volumes tab in Share and Storage Management, except for a system or boot volume.

Caution

When you delete a volume, all data and configuration information for the volume is also deleted, including all shared folders. If the volume contains any data, make certain that it has been appropriately backed up before deleting the volume.

Membership in **Administrators**, or equivalent, is the minimum required to complete this procedure.

To delete a volume

1. On the **Volumes** tab, click the volume that you want to delete.
2. In the **Actions** pane, click **Delete**.
3. In the **Delete Volume** dialog box, review the warning and, if you still want to continue, select **Yes**, and then click **OK**.

Detaching an Amazon EBS Volume from an Instance

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-detaching-volume.html>

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.

This example unmounts the volume and then explicitly detaches it from the instance. This is useful when you want to terminate an instance or attach a volume to a different instance.

Note that you can reattach a volume that you detached (without unmounting it), but it might not get the same mount point and the data on the volume might be out of sync if there were writes to the volume in progress when it was detached.

To detach an EBS volume using the console

1. Use the following command to unmount the /dev/sdh device.
[ec2-user ~]\$ **umount -d /dev/sdh**
2. Open the Amazon EC2 console at <https://console.aws.amazon.com/ec2/>.
3. In the navigation pane, choose **Volumes**.
4. Select a volume and choose **Actions, Detach Volume**.
5. In the confirmation dialog box, choose **Yes, Detach**.

Deleting an Amazon EBS Volume

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-deleting-volume.html>

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

To delete a volume, it must be in the available state (not attached to an instance).

To delete an EBS volume using the console

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. In the navigation pane, choose **Volumes**.
3. Select a volume and choose **Actions, Delete Volume**.
4. In the confirmation dialog box, choose **Yes, Delete**.

Amazon EC2 Instance Store

Launch an Instance Store Backed EC2 Instance in a VPC

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/launching-instance.html>

HINT This lab **CANNOT** be done in Free Tier. You can review procedure only, or you execute, but you will be charged.

An instance is a virtual server in the AWS cloud. You launch an instance from an Amazon Machine Image (AMI). The AMI provides the operating system, application server, and applications for your instance.

When you sign up for AWS, you can get started with Amazon EC2 for free using the [AWS Free Tier](#). You can either leverage the free tier to launch and use a micro instance for free for 12 months.

To launch an instance

1. Open the **Amazon EC2 console** at <https://console.aws.amazon.com/ec2/>.
2. In the navigation bar at the top of the screen, the current region is displayed. **Select the region** for the instance. This choice is important because some Amazon EC2 resources can be shared between regions, while others can't. Select the region that meets your needs. For more information, see [Resource Locations](#).
3. From the Amazon EC2 console dashboard, choose **Launch Instance**.
4. On the Choose an **Amazon Machine Image (AMI)** page, choose an AMI and then choose **Select**.
(Ex. **amzn-ami-pv-2012.03.2.x86_64-s3** - ami-0078da69)
5. On the **Choose an Instance Type** page, select the hardware configuration and size of the instance to launch. Larger instance types have more CPU and memory.
To remain eligible for the free tier, choose the **m1.small** instance type.
Choose Next: Configure Instance Details.
6. On the **Configure Instance Details** page, change the following settings as necessary (expand Advanced Details to see all the settings), and then choose **Next: Add Storage:**
(Ex.
Number of instances: 1
Network: Select oawsVPC
Subnet: Select subnet-us-east-1a-10.0.1.0/24
Auto-assign Public IP: Select Use subnet setting (Enable)
)
 - **Number of instances:** Enter the number of instances to launch.

- **Purchasing option:** Select Request Spot instances to launch a Spot instance. into EC2-Classic:
 - **Network:** Select the VPC, or to **create a new VPC**.
 - **Subnet:** Select the subnet into which to launch your instance.
 - **Auto-assign Public IP:** Specify whether your instance receives a public IPv4 address. By default, instances in a default subnet receive a public IPv4 address and instances in a nondefault subnet do not. You can select Enable or Disable to override the subnet's default setting.
 - **Auto-assign IPv6 IP:** Specify whether your instance receives an IPv6 address from the range of the subnet. Select Enable or Disable to override the subnet's default setting. This option is only available if you've associated an IPv6 CIDR block with your VPC and subnet.
 - **IAM role:** Select an AWS Identity and Access Management (IAM) role to associate with the instance.
 - **Shutdown behavior:** Select whether the instance should stop or terminate when shut down.
 - **Enable termination protection:** Select this check box to prevent accidental termination.
 - **Monitoring:** Select this check box to enable detailed monitoring of your instance using Amazon CloudWatch. Additional charges apply.
 - **EBS-Optimized instance:** An Amazon EBS-optimized instance uses an optimized configuration stack and provides additional, dedicated capacity for Amazon EBS I/O. If the instance type supports this feature, select this check box to enable it. Additional charges apply.
 - **Tenancy:** If you are launching your instance into a VPC, you can choose to run your instance on isolated, dedicated hardware (Dedicated) or on a Dedicated host (Dedicated host). Additional charges may apply.
 - **Network interfaces:** If you selected a specific subnet, you can specify up to two network interfaces for your instance:
 - **Kernel ID:** (Only valid for paravirtual (PV) AMIs) Select Use default unless you want to use a specific kernel.
 - **RAM disk ID:** (Only valid for paravirtual (PV) AMIs) Select Use default unless you want to use a specific RAM disk. If you have selected a kernel, you may need to select a specific RAM disk with the drivers to support it.
 - **Placement group:** A placement group is a logical grouping for your cluster instances. Select an existing placement group, or create a new one. This option is only available if you've selected an instance type that supports placement groups.
- User data:** You can specify user data to configure an instance during launch, or to run a configuration script. To attach a file, select the As file option and browse for the file to attach.
7. On the **Add Storage** page, you can specify volumes to attach to the instance besides the volumes specified by the AMI (such as the root device volume). You can change the following options, then choose **Next: Add Tags** when you have finished:
 8. On the **Add Tags** page, **specify tags** for the instance by providing key and value combinations. Choose **Add another tag** to add more than one tag to your resource. **Choose Next: Configure Security Group** when you are done.
(Ex. Key = Name ; Value = oawsEC2IS)
 9. On the **Configure Security Group** page, use a security group to define firewall rules for your instance. These rules specify which incoming network traffic is delivered to your instance. All other traffic is ignored. **Select** or create **a security group** as follows, and then **choose Review and Launch**.
To select an existing security group:
Choose Select an existing security group. Your security groups are displayed.
Select a security group from the list.
(Ex. oawsPublic-SG)
 10. On the **Review Instance Launch** page, **check the details** of your instance, and make any necessary changes by choosing the appropriate **Edit** link.
When you are ready, **choose Launch**.

11. In the **Select an existing key pair or create a new key pair** dialog box, you can choose an existing key pair, or create a new one. For example, **choose Choose an existing key pair**, then select the key pair you created when getting set up.
To launch your instance, **select the acknowledgment check box**, then **choose Launch Instances**.
Important If you choose the Proceed without key pair option, you won't be able to connect to the instance unless you choose an AMI that is configured to allow users another way to log in.

Stop Your EC2 Instance Store Backed Instance

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html

You can stop and restart your instance if it has an Amazon EBS volume as its root device. The instance retains its instance ID, but can change as described in the Overview section.

When you stop an instance, we shut it down. We don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes.

Stopping and Starting Your Instances

You can start and stop your Amazon EBS-backed instance using the console or the command line.

By default, when you initiate a shutdown from an Amazon EBS-backed instance (using the shutdown, halt, or poweroff command), the instance stops. You can change this behavior so that it terminates instead.

To stop and start an Amazon EBS-backed instance using the console

1. In the navigation pane, **choose Instances**, and **select the instance**.
2. [EC2-Classic] If the instance has an associated Elastic IP address, write down the Elastic IP address and the instance ID shown in the details pane.
3. **Choose Actions**, **select Instance State**, and then **choose Stop**. If **Stop** is disabled, either the instance is already stopped or its root device is an instance store volume.

Warning When you stop an instance, the data on any instance store volumes is erased. Therefore, if you have any data on instance store volumes that you want to keep, be sure to back it up to persistent storage.

HINT Stop is disabled because your root device is an instance store volume. The point is ... Don't use instance store volumes for persistent 'Important' data.

Adding Instance Store Volumes to an Instance

URL Link <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/add-instance-store-volumes.html>

HINT This lab **CANNOT** be done in Free Tier. You can review procedure only, or you execute, but you will be charged.

When you launch an instance, the default block device mapping is provided by the specified AMI. If you need additional instance store volumes, you must add them to the instance as you launch it. Note that you can also omit devices specified in the AMI block device mapping.

To update the block device mapping for an instance using the console

1. Open the **Amazon EC2 console**.
2. From the dashboard, **choose Launch Instance**.
3. In **Step 1: Choose an Amazon Machine Image (AMI)**, **select the AMI** to use and **choose Select**.
4. Follow the wizard to **complete Step 1: Choose an Amazon Machine Image (AMI)**, **Step 2: Choose an Instance Type**, and **Step 3: Configure Instance Details**.

- 5 In **Step 4: Add Storage**, modify the existing entries as needed. For each instance store volume to add, click **Add New Volume**, select an instance store volume from **Type**, and select a device name from **Device**. The number of available instance store volumes depends on the instance type.

Volume Type ⓘ	Device ⓘ	Snapshot ⓘ	Size (GiB) ⓘ
Root	/dev/xvda	snap-037f1f9e6c8ea4d65	8
Instance Store 0	/dev/sdb	N/A	N/A

Add New Volume

- 6 **Complete the wizard** to launch the instance.

HINT If you did this experiment on a Instance with a root device type of ebs, then added additional Instance store volumes. You could mount and use the Instance Store volumes, but when the server stopped – ALL DATA would be lost.

Stop Your EBS Backed EC2 Instance with Attached Instance Store Volumes

URL Link: http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Stop_Start.html

You can stop and restart your instance if it has an Amazon EBS volume as its root device. The instance retains its instance ID, but can change as described in the Overview section.

When you stop an instance, we shut it down. We don't charge hourly usage for a stopped instance, or data transfer fees, but we do charge for the storage for any Amazon EBS volumes.

Prerequisites

Launch EC2 Instances with EBS Backed Root Volume and Additional Instance Store Volumes

SSH to EC2 Instance

List Available Disk Volumes

```
lsblk
```

Create file called ebs.file on EBS volume

```
cd /  
echo "EBS" > ebs.file  
ls /
```

Create file called InstanceStore.file on EBS volume

```
cd /media/ephemeral0  
echo "Instance Store" > InstanceStore.file  
ls /media/ephemeral0
```

Stopping and Starting Your Instances

You can start and stop your Amazon EBS-backed instance using the console or the command line.

By default, when you initiate a shutdown from an Amazon EBS-backed instance (using the shutdown, halt, or poweroff command), the instance stops. You can change this behavior so that it terminates instead.

To stop and start an Amazon EBS-backed instance using the console

4. In the navigation pane, choose **Instances**, and select the instance.
5. Choose **Actions**, select **Instance State**, and then choose **Stop**. If **Stop** is disabled, either the instance is already stopped or its root device is an instance store volume.
Warning When you stop an instance, the data on any instance store volumes is erased. Therefore, if you have any data on instance store volumes that you want to keep, be sure to back it up to persistent storage.
6. To restart the stopped instance, select the instance, choose **Actions**, select **Instance State**, and then choose **Start**.
In the confirmation dialog box, choose **Yes, Start**. It can take a few minutes for the instance to enter the running state.
7. SSH to Instance (**HINT**: Public IP will have changed)

```
ls  
ls /media/ephemeral0
```

HINT Your data 'InstanceStore.file' is not on instance store volume. Because Instance Stores are ephemeral and the data only last as long as the Instance is running (or restarted). The point is ... Don't use instance store volumes for persistent 'Important' data.

Elastic File System (EFS)

Create Your Amazon EFS File System

URL Link <http://docs.aws.amazon.com/efs/latest/ug/gs-step-two-create-efs-resources.html>

In this step, you create your Amazon EFS file system.

To create your Amazon EFS file system

1. Open the **Amazon EFS console** at <https://console.aws.amazon.com/efs/>.
2. Choose **Create File System**.
3. Choose your VPC from the **VPC** list.
4. Select the check boxes for all of the Availability Zones. Make sure that they all have the default subnets, automatic IP addresses, and the default security groups chosen. These are your mount targets.
5. Choose **Next Step**.
6. Name your file system, keep **general purpose** selected as your default performance mode, and choose **Next Step**.
7. Choose **Create File System**.
8. Choose your file system from the list and make a note of the **File system ID** value. You'll need this value for the next step.

Connect to Your Amazon EC2 Instance and Mount the Amazon EFS File System

URL Link <http://docs.aws.amazon.com/efs/latest/ug/gs-step-three-connect-to-ec2-instance.html>

You can connect to your Amazon EC2 instance from a computer running Windows or Linux. To connect to your Amazon EC2 instance and mount the Amazon EFS file system, you need the following information:

- The **Public DNS** name of the Amazon EC2 instance.
- The **File system ID** value for the mount target for your Amazon EFS file system.

To connect to your Amazon EC2 instance and mount the Amazon EFS file system

1. Connect to your Amazon EC2 instance.
2. After you've connected, install the NFS client. If you're using an Amazon Linux AMI or RedHat Linux AMI, install the NFS client with the following command.

```
$ sudo yum -y install nfs-utils
```

If you're using an Ubuntu AML, install the NFS client with the following command.

```
$ sudo apt-get -y install nfs-common
```

3. Make a directory for the mount point with the following command.

```
$ sudo mkdir efs
```

4. Mount the Amazon EFS file system to the directory that you created. Use the following command and replace the *file-system-id* and *aws-region* placeholders with your **File System ID** value and AWS Region, respectively.

```
$ sudo mount -t nfs4 -o  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2 file-system-  
id.efs.aws-region.amazonaws.com:/ efs
```

Note We recommend that you wait 90 seconds after creating a mount target before you mount the file system, as the DNS records propagate fully in the region.

5. Change directories to the new directory that you created with the following command.

```
$ cd efs
```

6. Make a subdirectory and change the ownership of that subdirectory to your EC2 instance user. Then, navigate to that new directory with the following commands.

```
$ sudo mkdir getting-started  
$ sudo chown ec2-user getting-started  
$ cd getting-started
```

7. Create a text file with the following command.

```
$ touch test-file.txt
```

8. List the directory contents with the following command.

```
$ ls -al
```

As a result, the following file is created.

```
-rw-rw-r-- 1 ec2-user ec2-user 0 Aug 15 15:32 test-file.txt
```

To stop using an EFS volume on Linux

1. Unmount the file system

```
[ec2-user ~]$ sudo umount mountpoint
```

[Troubleshooting Amazon EFS](http://docs.aws.amazon.com/efs/latest/ug/troubleshooting.html)

URL Link: <http://docs.aws.amazon.com/efs/latest/ug/troubleshooting.html>

File System Mount Using DNS Name Fails

A file system mount that is using a DNS name fails. The following code shows an example.

```
$ sudo mount -t nfs4 -o  
nfsvers=4.1,rsize=1048576,wsiz=1048576,hard,timeo=600,retrans=2 file-  
system-id.efs.aws-region.amazonaws.com:/ mnt  
mount.nfs4: Failed to resolve server file-system-id.efs.aws-  
region.amazonaws.com:  
Name or service not known.
```

Action to Take

Check your VPC configuration. If you are using a custom VPC, you need to make sure DNS settings are enabled.

To specify a DNS name in the mount command, you must do the following:

- Ensure that there's an Amazon EFS mount target in the same Availability Zone as the Amazon EC2 instance.

- Connect your Amazon EC2 instance inside an Amazon VPC configured to use the DNS server provided by Amazon.
- Ensure that the Amazon VPC of the connecting Amazon EC2 instance has DNS host names enabled.

Updating DNS Support for Your VPC

URL Link <http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/vpc-dns.html>

You can view and update the DNS support attributes for your VPC using the Amazon VPC console.

To describe and update DNS support for a VPC using the console

1. Open the **Amazon VPC console** at <https://console.aws.amazon.com/vpc/>.
2. In the navigation pane, choose **Your VPCs**.
3. Select the VPC from the list.
4. Review the information in the **Summary** tab.
 - DNS resolution **yes**
 - DNS hostnames **yes**
5. To update these settings, choose **Actions** and either **Edit DNS Resolution** or **Edit DNS Hostnames**. In the dialog box that opens, choose **Yes** or **No**, and **Save**.

Deleting an Amazon EFS File System

URL Link: <http://docs.aws.amazon.com/efs/latest/ug/manage-delete-fs.html>

File system deletion is a destructive action that cannot be undone. You will lose the file system and any data you have in it.

Important You should always unmount a file system before you delete it.

Using the Console

1. Open the **Amazon Elastic File System console** at <https://console.aws.amazon.com/efs/>.
2. Select the file system that you want to delete.
3. Choose **Action** and then choose **Delete File System**.
4. In **Permanently Delete File System** confirmation box, type the file system ID and then choose **Delete File System**.

The console simplifies the file deletion for you. First it deletes the associated mount targets, and then it deletes the file system.

Relational Database Service (RDS)

Enabling Automated Backups

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithAutomatedBackups.html

If your DB instance doesn't have automated backups enabled, you can enable them at any time. You enable automated backups by setting the backup retention period to a positive non-zero value. When automated backups are enabled, an outage occurs and a backup is immediately created.

In this example, you enable automated backups for a DB instance by setting the backup retention period to a positive non-zero value.

AWS Management Console

To enable automated backups immediately

1. Sign in to the AWS Management Console and open the **Amazon RDS console** at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, choose **DB Instances**, and then select the DB instance that you want to modify.
3. Choose **Instance Actions**, and then choose **Modify**. The **Modify DB Instance** window appears.
4. For **Backup Retention Period**, choose a positive non-zero value, for example **1**.
5. Select **Apply Immediately**.
6. Choose **Continue**.
7. On the confirmation page, choose **Modify DB Instance** to save your changes and disable automated backups.

Disabling Automated Backups

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithAutomatedBackups.html

You may want to temporarily disable automated backups in certain situations; for example, while loading large amounts of data.

Important We highly discourage disabling automated backups because it disables point-in-time recovery. Disabling automatic backups for a DB instance deletes all existing automated backups for the instance. If you disable and then re-enable automated backups, you are only able to restore starting from the time you re-enabled automated backups.

In this example, you disable automated backups for a DB instance by setting the backup retention parameter to 0.

AWS Management Console

To disable automated backups immediately

1. Sign in to the AWS Management Console and open the Amazon RDS console at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, choose **DB Instances**, and then select the DB instance that you want to modify.
3. Choose **Instance Actions**, and then choose **Modify**. The **Modify DB Instance** window appears.
4. For **Backup Retention Period**, choose **0**.
5. Select **Apply Immediately**.
6. Choose **Continue**.
7. On the confirmation page, choose **Modify DB Instance** to save your changes and disable automated backups.

Restoring a DB Instance to a Specified Time

URL Link http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIT.html

To restore a DB instance to a specified time

1. Sign in to the AWS Management Console and open the **Amazon RDS console** at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, click **DB Instances**.
3. Click **Instance Actions**, and then click **Restore To Point In Time**. The **Restore DB Instance** window appears.
4. Click on the **Use Custom Restore Time** radio button.
5. Enter the date and time that you wish to restore to in the **Use Custom Restore Time** text boxes.
6. Type the name of the restored DB instance in the **DB Instance Identifier** text box.
7. Click the **Launch DB Instance** button.

Hint Restoring a DB Instance creates a new database with a new endpoint.

Creating a DB Snapshot

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_CreateSnapshot.html

Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. Creating this DB snapshot on a Single-AZ DB instance results in a brief I/O suspension that typically lasting no more than a few minutes. Multi-AZ DB instances are not affected by this I/O suspension since the backup is taken on the standby.

When you create a DB snapshot, you need to identify which DB instance you are going to back up, and then give your DB snapshot a name so you can restore from it later.

In this example, you create a DB snapshot for a DB instance.

AWS Management Console

To create a DB snapshot

1. Sign in to the AWS Management Console and open the **Amazon RDS console** at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, **click DB Instances**.
3. **Click Instance Actions**, and then **click Take DB Snapshot**. The **Take DB Snapshot** window appears.
4. **Type the name** of the snapshot in the **Snapshot Name** text box.
5. **Click Yes, Take Snapshot**.

Restoring From a DB Snapshot

URL Link http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RestoreFromSnapshot.html

To restore a DB instance from a DB snapshot

1. Sign in to the AWS Management Console and open the **Amazon RDS console** at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, **choose Snapshots**.
3. **Choose the DB snapshot** that you want to restore from.
4. **Choose Restore Snapshot**. The Restore DB Instance window appears.
5. **Type the name** of the restored DB instance in the **DB Instance Identifier** text box.
6. **Choose Restore DB Instance**.
7. Sign in to the AWS Management Console and open the **Amazon VPC console** at <https://console.aws.amazon.com/vpc/>.
8. In the navigation pane, **choose Security Groups**.
9. Select the security group that you want to use for your DB instances. If you need to **add rules** to link the security group to a security group for an EC2 instance.

Copying a DB Snapshot by Using the AWS Management Console

URL Link http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_CopySnapshot.html

This procedure works for copying encrypted or unencrypted DB snapshots, in the same region or across regions.

To copy a DB snapshot

1. Sign in to the AWS Management Console and open the **Amazon RDS console** at <https://console.aws.amazon.com/rds/>.
2. In the navigation pane, **choose Snapshots**.
3. **Select the check box** for the DB snapshot you want to copy.

4. Choose **Snapshot Actions**, and then choose **Copy Snapshot**.
5. (Optional) To copy the DB snapshot to a different region, choose that region for **Destination Region**.
Note The destination region must have the same database engine version available as the source region.
6. Type the name of the DB snapshot copy in **New DB Snapshot Identifier**.
7. To copy tags and values from the snapshot to the copy of the snapshot, choose **Copy Tags**.
8. Choose one of the following options for **Enable Encryption**.
 - If the DB snapshot isn't encrypted and you don't want to encrypt the copy, choose **No** for **Enable Encryption**.
 - If the DB snapshot isn't encrypted but you want to encrypt the copy, choose **Yes** for **Enable Encryption**, and then specify the KMS key identifier to use to encrypt the DB snapshot copy for **Master Key**.
9. If the DB snapshot being copied is encrypted, you must encrypt the copy, so **Enable Encryption** is already set to **Yes**. Specify the KMS key identifier to use to encrypt the DB snapshot copy for **Master Key**.
10. Choose **Copy Snapshot**.

WordPress Web Server

Upload Image to WordPress Media Library

To Upload a File in a Post

- 1 On the **Dashboard** menu, click **Posts**, and then click **Add New** to display the "Add New Post" page.
 - 2 On the **Upload/Insert** menu, click the icon for the type of file you want to upload and the "Add media files from your computer" page will appear.
 - 3 Click the **Select Files** button.
 - 4 In the dialog box, select the file you want to upload.
 To select multiple files, hold down the SHIFT key (for PC users) or the COMMAND key (for Macintosh users).
 - 5 Click **Open**.
 - 6 When your file uploads, a field appears. At the bottom of the field, click the **Insert into Post** button.
- Note:** If you are having problems uploading files with the default Flash uploader, you may want to use the [Browser uploader](#) instead.

To View File on WordPress Web Server

SSH to WordPress Instance

```
sudo su
cd /vaw/www/html/wp-content
ls uploads/
```

Hint You should see a folder with current year. Navigate through sub-directories (i.e. cd directory) until you locate uploaded file.

To Upload a File for Later Use

- 1 On the **Dashboard** menu, click **Media** and then click **Add New** to display the "Upload New Media" page.
- 2 Click the **Select Files** button to open a dialog box.
- 3 In the dialog box, select the file you want to upload.
 To select multiple files, hold down the SHIFT key (for PC users) or the COMMAND key (for Macintosh users).
- 4 Click the **Open** button.
- 5 When the upload is complete, a field with your file details appears. Below the field, click **Save all changes**.

Note: If the file does not open, then the file type is not supported, the chosen format may not match the file's true format or the file may be damaged.

Copy WordPress Code and Content to S3 bucket

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/delete-or-empty-bucket.html#empty-bucket-console>

The Amazon S3 console supports emptying your bucket provided that the bucket contains less than 100,000 objects.

To empty a bucket

In the Amazon S3 console, open the context (right-click) menu on the bucket and choose Empty Bucket.

To copy configuration files and content using AWS CLI

SSH to WordPress Web Server

```
sudo su
cd /var/www/html
aws s3 cp /var/www/html s3://<bucket-name>/code/
aws s3 cp /var/www/html/wp-content/uploads s3://<bucket-name>/uploads/
```

Configure http for URL Re-Write

URL Link: <http://httpd.apache.org/docs/2.0/misc/rewriteguide.html>

URL Re-Write is done by configuring the httpd daemon conf file to allow OverRide. In simple terms, we can store content, such as images, in a different location than on the Web Server, and users browsing our website will be redirected. We will use in our lab to re-direct traffic to CloudFront CDN for image content.

To enable URL ReWrite

```
cd /etc/httpd/conf
cp httpd.conf httpd.conf.backup
rm -rf httpd.conf
wget https://s3.amazonaws.com/www.ouraws.com/wordpress/config/httpd.conf
cd /var/www/html
wget https://s3.amazonaws.com/www.ouraws.com/wordpress/config/htaccess
mv htaccess .htaccess
nano .htaccess
```

Replace <CloudFront Domain Name> with your CDN Domain Name

CDN Domain Name can be located in CloudFront – Distribution – Domain Name

Save and Exit File

To validate URL ReWrite

Restart http services

Navigate to website, and select uploaded file in post. Click copy Link, and paste in new browser window.

Observer URL

Hint URL will be serving off of CloudFront CDN not WordPress Web Server

WordPress Database Settings

URL Link https://codex.wordpress.org/Database_Description

During our lessons, we may observe an issue where the site fails to load properly. If this occurs, check the wp_options table to ensure site is not linked to IP of EC2 instance (which may change).

To connect and update to WordPress Database

1. SSH to EC2 WordPress Web Server Instance.

2. Identify 'siteurl' and 'home' configuration

```
sudo su
yum install mysql -y
mysql -h <DB EndPoint> -u <User> -p
    Provide password when prompted
mysql> use <DBName>;
mysql> show tables;
mysql> describe wp_options;
mysql> select * from wp_options where option_name = 'siteurl';
mysql> select * from wp_options where option_name = 'home';
```

3. Update table with Route 53 Alias Record Set

```
mysql> update wp_options set option_value = 'http://www.ouraws.com' where
option_id = 1;
mysql> update wp_options set option_value = 'http://www.ouaws.com' where
option_id = 2;
```

Cleanup!

Delete / Stop All Billable Resources

Note: You only pay for what you use. **Why pay if you are not using it? Shut it down. Clean up.** You'll build what you need in the next lab.

To clean up after lesson

Terminate All EC2 Instances

Delete Any MySQL DB Instance Snapshots You Created

Delete Any Additional RDS DB Instances You Created

Delete Any EFS File Systems You Created

Delete Any AMIs You Created

Delete Any EBS Volume Snapshots You Created

Delete Any EBS Volumes You Created

Restore Any Bucket Configurations You Changed

HINT While we recommend you practice creating and deleting resources, we will need some resources for future lessons. Practice deleting, then rebuild to the required configuration before next lesson.

Environment Configuration

At the end of this lab, your environment should be configured as follows

Lesson 4

Simple Storage Service (S3) – 2 Buckets (Source/Replication)

S3 Bucket Versioning Enabled

Lifecycle Configuration Rule Added to S3 Bucket

Cross Region Replication Enabled to Second Region

Elastic Compute Cloud (EC2) – 1 Amazon Linux Instance (WordPress Web Server)

Snapshot of WordPress root EBS Volume Created

AMI of WordPress Server Created

Relational Database Services (RDS) – 1 RDS MySQL DB Instance

Automated Backups Enable and Retention Period Set to 1 Day

Snapshot of MySQL DB Instance Created

Possibly Snapshot of MySQL DB Instance Copied to Another Region

WordPress Web Server – 1 WordPress Web Server

Media File Uploaded to Post

WordPress Configuration Copied to S3 Bucket Code Folder

WordPress Uploads Copied to S3 Bucket Uploads Folder

URL Re-write Enabled and pointing guests to CDN for Media Uploads

Word Press Database table records 'urlsite' and 'home' updated.

Hint If you are confused, don't worry. We'll try and explain at a high level what we have achieved. In this Lab, we have protected three components of our infrastructure:

- **EC2 Instance and Application Configuration**
We have create both a Snapshot and an AMI of our WordPress Server. This includes the OS, the applications we have installed (http, php, php-mysql and WordPress)
- **WordPress Application Code**
We have coped the contents of our /var/www/html directory on our WordPress server to our S3 bucket. This directory includes all of the WordPress configuration (i.e. wp-content) that may change as we update our site
- **WordPress Content**
We have copied the contents of our /var/www/html/wp-content/uploads to our S3 bucket. This directory contains any files we upload to WordPress
- **URL Re-Write (Re-Direct)**
We have changed the configuration of Apache http to redirect request to wp-content/uploads to our CloudFront CDN.
... and improved the customer experience by adding low latency retrieval of Media Files.
- **Wordpress Database Settings**
We have changed the wordpress database to use our domain name, and not the IP of the instance which could change.

Still Confused?

S3 is a more durable and available store of data than our EC2 Instance EBS Root volume. Yes, we can restore our Web Server using our AMI or Snapshot, but we will lose any content since the last time we copied to S3. Additionally, we are leveraging CDN Edge networks to server media to customers who are located at a distance to our location. Lastly, We changed data in our database table to make our website more resilient.

Knowledge Base









AWS Storage Services Overview

URL Link <https://d0.awsstatic.com/whitepapers/Storage/AWS%20Storage%20Services%20Whitepaper-v9.pdf>

HINT This is an important AWS White Paper. If you intend to take any of the AWS Certified Exams, this is a **MUST READ**. We will work with most of the services in this lesson, but not all. Some of the topics are strictly Business or Enterprise related, and beyond the scope of our ability to experiment (i.e. we don't have or don't wish to spend the money to acquire the equipment to support testing). However, we will cover in later theoretical lessons.

Introduction

Amazon Web Services (AWS) provides low-cost data storage with high durability and availability. AWS offers storage choices for backup, archiving, and disaster recovery use cases and provides block, file, and object storage. In this whitepaper, we examine the following AWS Cloud storage services and features.

	Amazon Simple Storage Service (Amazon S3)	A service that provides scalable and highly durable object storage in the cloud.
	Amazon Glacier	A service that provides low-cost highly durable archive storage in the cloud.
	Amazon Elastic File System (Amazon EFS)	A service that provides scalable network file storage for Amazon EC2 instances.
	Amazon Elastic Block Store (Amazon EBS)	A service that provides block storage volumes for Amazon EC2 instances.
	Amazon EC2 Instance Storage	Temporary block storage volumes for Amazon EC2 instances.
	AWS Storage Gateway	An on-premises storage appliance that integrates with cloud storage.
	AWS Snowball	A service that transports large amounts of data to and from the cloud.
	Amazon CloudFront	A service that provides a global content delivery network (CDN).

HINT The White Paper discusses each service in terms of the following:

- Usage Patterns
- Performance
- Durability and Availability
- Scalability and Elasticity
- Security
- Interfaces
- Cost Model

We will review most of the characteristics here, and discuss the remaining information in later lessons.

Amazon S3

Amazon Simple Storage Service (Amazon S3) provides developers and IT teams secure, durable, highly scalable object storage at a very low cost. You can store and retrieve any amount of data, at any time, from anywhere on the web through a simple web service interface. You can write, read, and delete objects containing from 1 byte to 5 TB of data. Amazon S3 is highly scalable, allowing concurrent read or write access to data by many separate clients or application threads.

Amazon S3 offers a range of storage classes designed for different use cases including the following:

- Amazon S3 Standard, for general-purpose storage of frequently accessed data
- Amazon S3 Standard-Infrequent Access (Standard-IA), for long-lived, but less frequently accessed data
- Amazon Glacier, for low-cost archival data Usage Patterns

Usage Patterns

There are four common usage patterns for Amazon S3.

- First, Amazon S3 is used to store and distribute static web content and media.
- Second, Amazon S3 is used to host entire static websites.
- Third, Amazon S3 is used as a data store for computation and large-scale analytics, such as financial transaction analysis, clickstream analytics, and media transcoding.
- Finally, Amazon S3 is often used as a highly durable, scalable, and secure solution for backup and archiving of critical data.

Amazon S3 doesn't suit all storage situations. The following table presents some storage needs for which you should consider other AWS storage options.

Storage Need	Solution	AWS Services
File system	Amazon S3 uses a flat namespace and isn't meant to serve as a standalone, POSIX-compliant file system. Instead, consider using Amazon EFS as a file system.	Amazon EFS
Structured data with query	Amazon S3 doesn't offer query capabilities to retrieve specific objects. When you use Amazon S3 you need to know the exact bucket name and key for the files you want to retrieve from the service. Amazon S3 can't be used as a database or search engine by itself. Instead, you can pair Amazon S3 with Amazon DynamoDB, Amazon CloudSearch, or Amazon Relational Database Service (Amazon RDS) to index and query metadata about Amazon S3 buckets and objects.	Amazon DynamoDB Amazon RDS Amazon CloudSearch
Rapidly changing data	Data that must be updated very frequently might be better served by storage solutions that take into account read and write latencies, such as Amazon EBS volumes, Amazon RDS, Amazon DynamoDB, Amazon EFS, or relational databases running on Amazon EC2.	Amazon EBS Amazon EFS Amazon DynamoDB Amazon RDS
Archival data	Data that requires encrypted archival storage with infrequent read access with a long recovery time objective (RTO) can be stored in Amazon Glacier more cost-effectively.	Amazon Glacier
Dynamic website hosting	Although Amazon S3 is ideal for static content websites, dynamic websites that depend on database interaction or use server-side scripting should be hosted on Amazon EC2 or Amazon EFS.	Amazon EC2 Amazon EFS

Performance

In scenarios where you use Amazon S3 from within Amazon EC2 in the same Region, access to Amazon S3 from Amazon EC2 is designed to be fast. Amazon S3 is also designed so that server-side latencies are insignificant relative to Internet latencies. In addition, Amazon S3 is built to scale storage, requests, and numbers of users to support an extremely large number of web-scale applications.

To improve the upload performance of large objects (typically over 100 MB), Amazon S3 offers a [multipart upload](#) command to upload a single object as a set of parts. After all parts of your object are uploaded, Amazon S3 assembles these parts and creates the object. Using multipart upload, you can get improved throughput and quick recovery from any network issues. Another benefit of using multipart upload is that you can upload multiple parts of a single object in parallel and restart the upload of smaller parts instead of restarting the upload of the entire large object.

To speed up access to relevant data, many developers pair Amazon S3 with a database such as Amazon RDS. In these scenarios, Amazon S3 stores the actual information, and database serves as the repository for associated metadata (for example, the object name, size, keywords, and so on). Metadata in the database can easily be indexed and queried, making it very efficient to locate an object's reference by using a search engine or a database query. This result can be used to pinpoint and retrieve the object itself from Amazon S3.

Durability and Availability

Amazon S3 Standard storage and Standard-IA storage provide high levels of data durability and availability by automatically and synchronously storing your data across both multiple devices and multiple facilities within your selected geographical region. Error correction is built-in, and there are no single points of failure. Amazon S3 is designed to sustain the concurrent loss of data in two facilities, making it very well suited to serve as the primary data storage for mission-critical data. In fact, Amazon S3 is designed for 99.99999999 percent (11 nines) durability per object and 99.99 percent availability over a one-year period.

Interfaces

Amazon S3 provides standards-based REST web service application program interfaces (APIs) for both management and data operations. These APIs allow Amazon S3 objects to be stored in uniquely named buckets (top-level folders). Each object must have a unique object key (file name) that serves as an identifier for the object within that bucket. Although Amazon S3 is a web-based object store with a flat naming structure rather than a traditional file system, you can easily emulate a file system hierarchy (folder1/folder2/file) in Amazon S3 by creating object key names that correspond to the full path name of each file.

Most developers building applications on Amazon S3 use a higher-level toolkit or software development kit (SDK) that wraps the underlying REST API. AWS SDKs are available for Android, Browser, iOS, Java, .NET, Node.js, PHP, Python, Ruby, and Go. The integrated AWS Command Line Interface (AWS CLI) also provides a set of high-level, Linux-like Amazon S3 file commands for common operations, such as ls, cp, mv, sync, and so on. Using the AWS CLI for Amazon S3, you can perform recursive uploads and downloads using a single folder-level Amazon S3 command and also perform parallel transfers. You can also use the AWS CLI for command-line access to the low-level Amazon S3 API. Using the AWS Management Console, you can easily create and manage Amazon S3 buckets, upload and download objects, and browse the contents of your S3 buckets using a simple web-based user interface.

Cost Model

With Amazon S3, you pay only for the storage you actually use. There is no minimum fee and no setup cost. Amazon S3 Standard has three pricing components: storage (per GB per month), data transfer in or out (per GB per month), and requests (per thousand requests per month). For new customers, AWS provides [the AWS Free](#)

[Tier](#), which includes up to 5 GB of Amazon S3 storage, 20,000 get requests, 2,000 put requests, and 15 GB of data transfer out each month for one year, for free.

Amazon Glacier

Amazon Glacier is an extremely low-cost storage service that provides highly secure, durable, and flexible storage for data archiving and online backup. With Amazon Glacier, you can reliably store your data for as little as \$0.007 per gigabyte per month.

You store data in Amazon Glacier as archives. An archive can represent a single file, or you can combine several files to be uploaded as a single archive.

Retrieving archives from Amazon Glacier requires the initiation of a job. You organize your archives in vaults.

Amazon Glacier is designed for use with other Amazon web services. You can seamlessly move data between Amazon Glacier and Amazon S3 using S3 data lifecycle policies.

Usage Patterns

Organizations are using Amazon Glacier to support a number of use cases. These use cases include archiving offsite enterprise information, media assets, and research and scientific data, and also performing digital preservation and magnetic tape replacement.

Amazon Glacier doesn't suit all storage situations. The following table presents a few storage needs for which you should consider other AWS storage options.

Storage Need	Solution	AWS Services
Rapidly changing data	Data that must be updated very frequently might be better served by a storage solution with lower read/write latencies, such as Amazon EBS, Amazon RDS, Amazon EFS, Amazon DynamoDB, or relational databases running on Amazon EC2.	Amazon EBS Amazon RDS Amazon EFS Amazon DynamoDB Amazon EC2
Immediate access	Data stored in Amazon Glacier is not available immediately. Retrieval jobs typically require 3–5 hours to complete, so if you need immediate access to your object data, Amazon S3 is a better choice.	Amazon S3

Performance

Amazon Glacier is a low-cost storage service designed to store data that is infrequently accessed and long-lived. Amazon Glacier retrieval jobs typically complete in 3 to 5 hours.

You can improve the upload experience for larger archives by using [multipart upload](#) for archives up to about 40 TB (the single archive limit). You can upload separate parts of a large archive independently, in any order and in parallel, to improve the upload experience for larger archives. You can even perform [range retrievals](#) on archives stored in Amazon Glacier by specifying a range or portion of the archive. Specifying a range of bytes for a retrieval can help control bandwidth costs, manage your data downloads, and retrieve a targeted part of a large archive.

Durability and Availability

Amazon Glacier is designed to provide average annual durability of 99.99999999 percent (11 nines) for an archive. The service redundantly stores data in multiple facilities and on multiple devices within each facility. To

increase durability, Amazon Glacier synchronously stores your data across multiple facilities before returning SUCCESS on uploading an archive. Unlike traditional systems, which can require laborious data verification and manual repair, Amazon Glacier performs regular, systematic data integrity checks and is built to be automatically self-healing.

Interfaces

There are two ways to use Amazon Glacier, each with its own interfaces. The Amazon Glacier API provides both management and data operations.

First, Amazon Glacier provides a native, standards-based REST web services interface. This interface can be accessed using the Java SDK or the .NET SDK. You can use the AWS Management Console or Amazon Glacier API actions to create vaults to organize the archives in Amazon Glacier.

Second, Amazon Glacier can be used as a storage class in Amazon S3 by using object lifecycle management that provides automatic, policy-driven archiving from Amazon S3 to Amazon Glacier. You simply set one or more lifecycle rules for an Amazon S3 bucket, defining what objects should be transitioned to Amazon Glacier and when.

Retrieval puts a copy of the retrieved object in Amazon S3 Reduced Redundancy Storage (RRS) for a specified retention period. The original archived object remains stored in Amazon Glacier.

Note that when using Amazon Glacier as a storage class in Amazon S3 you use the Amazon S3 API, and when using “native” Amazon Glacier you use the Amazon Glacier API. For example, objects archived to Amazon Glacier using Amazon S3 lifecycle policies can only be listed and retrieved by using the Amazon S3 API or the Amazon S3 console. You can’t see them as archives in an Amazon Glacier vault.

Cost Model

With Amazon Glacier, you pay only for what you use and there is no minimum fee. In normal use, Amazon Glacier has three pricing components: storage (per GB per month), data transfer out (per GB per month), and requests (per thousand UPLOAD and RETRIEVAL requests per month).

Note that Amazon Glacier is designed with the expectation that retrievals are infrequent and unusual, and data will be stored for extended periods of time. You can retrieve up to 5 percent of your average monthly storage (prorated daily) for free each month. If you retrieve more than this amount of data in a month, you are charged an additional (per GB) retrieval fee. A prorated charge (per GB) also applies for items deleted prior to 90 days’ passage.

Amazon EBS

Amazon Elastic Block Store (Amazon EBS) volumes provide durable block-level storage for use with EC2 instances. Amazon EBS volumes are network-attached storage that persists independently from the running life of a single EC2 instance. After an EBS volume is attached to an EC2 instance, you can use the EBS volume like a physical hard drive, typically by formatting it with the file system of your choice and using the file I/O interface provided by the instance operating system. Most Amazon Machine Images (AMIs) are backed by Amazon EBS, and use an EBS volume to boot EC2 instances. You can also attach multiple EBS volumes to a single EC2 instance. Note, however, that any single EBS volume can be attached to only one EC2 instance at any time.

EBS also provides the ability to create point-in-time snapshots of volumes, which are stored in Amazon S3. These snapshots can be used as the starting point for new EBS volumes and to protect data for long-term durability. The same snapshot can be used to instantiate as many volumes as you want. These snapshots can be copied across AWS Regions, making it easier to leverage multiple AWS Regions for geographical expansion, data center

migration, and disaster recovery. Sizes for EBS volumes range from 1 GiB to 16 TiB, depending on the volume type, and are allocated in 1 GiB increments.

Usage Patterns

Amazon EBS is meant for data that changes relatively frequently and needs to persist beyond the life of EC2 instance. Amazon EBS is well-suited for use as the primary storage for a database or file system, or for any application or instance (operating system) that requires direct access to raw block-level storage. Amazon EBS provides a range of options that allow you to optimize storage performance and cost for your workload. These options are divided into two major categories: solid-state drive (SSD)-backed storage for transactional workloads such as databases and boot volumes (performance depends primarily on IOPS) and hard disk drive (HDD)-backed storage for throughput-intensive workloads such as big data, data warehouse, and log processing (performance depends primarily on MB/s).

Amazon EBS doesn't suit all storage situations. The following table presents some storage needs for which you should consider other AWS storage options.

Storage Need	Solution	AWS Services
Temporary storage	Consider using local instance store volumes for needs such as scratch disks, buffers, queues, and caches.	Amazon Local Instance Store
Multi-instance storage	Amazon EBS volumes can only be attached to one EC2 instance at a time. If you need multiple EC2 instances accessing volume data at the same time, consider using Amazon EFS as a file system.	Amazon EFS
Highly durable storage	If you need very highly durable storage, use S3 or Amazon EFS. Amazon S3 Standard storage is designed for 99.99999999 percent (11 nines) annual durability per object. You can even decide to take a snapshot of the EBS volumes. Such a snapshot then gets saved in Amazon S3, thus providing you the durability of Amazon S3. For more information on EBS durability, see the Durability and Availability section. EFS is designed for high durability and high availability, with data stored in multiple Availability Zones within an AWS Region.	Amazon S3 Amazon EFS
Static data or web content	If your data doesn't change that often, Amazon S3 might represent a more cost-effective and scalable solution for storing this fixed information. Also, web content served out of Amazon EBS requires a web server running on Amazon EC2; in contrast, you can deliver web content directly out of Amazon S3 or from multiple EC2 instances using Amazon EFS.	Amazon S3 Amazon EFS

Performance

As described previously, Amazon EBS provides a range of volume types that are divided into two major categories: SSD-backed storage volumes and HDD-backed storage volumes. SSD-backed storage volumes offer great price/performance characteristics for random small block workloads, such as transactional applications, whereas HDD-backed storage volumes offer the best price/performance characteristics for large block sequential workloads. You can attach and stripe data across multiple volumes of any type to increase the I/O performance available to your Amazon EC2 applications.

General Purpose SSD (gp2) 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 baseline performance of 3 IOPS/GiB up to a maximum of 10,000 IOPS (at 3,334 GiB). The gp2 volumes can range in size from 1 GiB to 16 TiB.

Provisioned IOPS SSD (io1) volumes are designed to deliver predictable high performance for I/O-intensive workloads with small I/O size where the dominant performance attribute is IOPS, such as database workloads that are sensitive to storage performance and consistency in random access I/O throughput. The io1 volumes can range in size from 4 GiB to 16 TiB, and you can provision up to 20,000 IOPS per volume.

Throughput Optimized HDD (st1) volumes are ideal for frequently accessed, throughput-intensive workloads with large datasets and large I/O sizes, where the dominant performance attribute is throughput (MiB/s), such as streaming workloads, big data, data warehouse, log processing, and ETL workloads. The st1 volumes can't be used as boot volumes.

Cold HDD (sc1) volumes provide the lowest cost per GiB of all EBS volume types. These are ideal for infrequently accessed workloads with large, cold datasets with large I/O sizes, where the dominant performance attribute is throughput (MiB/s). The sc1 volumes can't be used as boot volumes.

Because all EBS volumes are network-attached devices, other network I/O performed by an EC2 instance, as well as the total load on the shared network, can affect the performance of individual EBS volumes. To enable your EC2 instances to maximize the performance of EBS volumes, you can launch selected EC2 instance types as EBS-optimized instances. Most of the latest generation EC2 instances (m4, c4, x1, and p2) are EBS-optimized by default. EBS-optimized instances deliver dedicated throughput between Amazon EC2 and Amazon EBS, with speeds between 500 Mbps and 10,000 Mbps depending on the instance type.

Using Amazon EC2 with Amazon EBS, you can take advantage of many of the same disk performance optimization techniques that you do with on-premises servers and storage. For example, by attaching multiple EBS volumes to a single EC2 instance, you can partition the total application I/O load by allocating one volume for database log data, one or more volumes for database file storage, and other volumes for file system data. Each separate EBS volume can be configured as EBS General Purpose (SSD), Provisioned IOPS (SSD), Throughput Optimized (HDD), or Cold (HDD) as needed. Some of the best price/performance balanced workloads take advantage of different volume types on a single EC2 instance. Alternatively, you can stripe your data across multiple similarly provisioned EBS volumes using RAID 0 (disk striping) or logical volume manager software, thus aggregating available IOPS, total volume throughput, and total volume size.

Durability and Availability

Amazon EBS volumes are designed to be highly available and reliable. EBS volume data is replicated across multiple servers in a single Availability Zone to prevent the loss of data from the failure of any single component. Taking snapshots of your EBS volumes increases the durability of the data stored on your EBS volumes. EBS snapshots are incremental, point-in-time backups, containing only the data blocks changed since the last snapshot. We recommend that you create snapshots of your EBS volumes to improve the durability of your data.

To maximize both durability and availability of Amazon EBS data, you should create snapshots of your EBS volumes frequently. (For application-consistent backups, we recommend briefly pausing any write operations to the volume, or unmounting the volume, while you issue the snapshot command. You can then safely continue to use the volume while the snapshot is pending completion.) Because an EBS volume is created in a particular Availability Zone, the volume will be unavailable if the Availability Zone itself is unavailable. A snapshot of a volume, however, is available across all of the Availability Zones within a Region, and you can use a snapshot to create one or more new EBS volumes in any Availability Zone in the region. EBS snapshots can also be copied from one Region to another, and can easily be shared with other user accounts. Thus, EBS snapshots provide an easy-to-use disk clone or disk image mechanism for backup, sharing, and disaster recovery.

Interfaces

Amazon offers a REST management API for Amazon EBS, as well as support for Amazon EBS operations within both the AWS SDKs and the AWS CLI. If you prefer to work with a graphical user interface, the AWS Management Console gives you all the capabilities of the API in a browser interface. Regardless of how you create your EBS volume, note that all storage is allocated at the time of volume creation, and that you are charged for this allocated storage even if you don't write data to it.

Amazon EBS doesn't provide a data API. Instead, Amazon EBS presents a block- device interface to the EC2 instance. That is, to the EC2 instance, an EBS volume appears just like a local disk drive. To write to and read data from EBS volumes, you use the native file system I/O interfaces of your chosen operating system.

Cost Model

As with other AWS services, with Amazon EBS you pay only for what you provision, in increments down to 1 GB. In contrast, hard disks come in fixed sizes and you pay for the entire size of the disk regardless of the amount you use or allocate. Amazon EBS pricing has three components: provisioned storage, I/O requests, and snapshot storage. Amazon EBS General Purpose (SSD), Throughput Optimized (HDD), and Cold (HDD) volumes are charged per GB- month of provisioned storage. Amazon EBS Provisioned IOPS (SSD) volumes are charged per GB-month of provisioned storage and per provisioned IOPS-month. For all volume types, Amazon EBS snapshots are charged per GB-month of data stored. An Amazon EBS snapshot copy is charged for the data transferred between Regions and for the standard Amazon EBS snapshot charges in the destination Region.

It's important to remember that for EBS volumes, you are charged for provisioned (allocated) storage, whether or not you actually use it. For Amazon EBS snapshots, you are charged only for storage actually used (consumed). Note that Amazon EBS snapshots are incremental so the storage used in any snapshot is generally much less than the storage consumed for an EBS volume.

Note that there is no charge for transferring information among the various AWS storage offerings (that is, an EC2 instance transferring information with Amazon EBS, Amazon S3, Amazon RDS, and so on) as long as the storage offerings are within the same AWS Region.

Amazon EC2 Instance Storage

Amazon EC2 instance store volumes (also called ephemeral drives) provide temporary block-level storage for many EC2 instance types. This storage consists of a preconfigured and pre-attached block of disk storage on the same physical server that hosts the EC2 instance for which the block provides storage. The amount of the disk storage provided varies by EC2 instance type. In the EC2 instance families that provide instance storage, larger instances tend to provide both more and larger instance store volumes.

Note that some instance types, such as the micro instances (t1, t2) and the Compute-optimized c4 instances, use EBS storage only with no instance storage provided. Note also that instances using Amazon EBS for the root device (in other words that boot from Amazon EBS) don't expose the instance store volumes by default.

Usage Patterns

In general, EC2 local instance store volumes are ideal for temporary storage of information that is continually changing, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers. This storage can only be used from a single EC2 instance during that instance's lifetime. Note that, unlike EBS volumes, instance store volumes cannot be detached or attached to another instance.

For high I/O and high storage, use EC2 instance storage targeted to these use cases. High I/O instances (the i2 family) provide instance store volumes backed by SSD and are ideally suited for many high-performance database workloads. High storage instances (the d2 family) support much higher storage density per EC2 instance and are ideally suited for applications that benefit from high sequential I/O performance across very large datasets.

Note that applications using instance storage for persistent data generally provide data durability through replication, or by periodically copying data to durable storage.

EC2 instance store volumes don't suit all storage situations. The following table presents some storage needs for which you should consider other AWS storage options.

Storage Need	Solution	AWS Services
Persistent storage	If you need persistent virtual disk storage similar to a physical disk drive for files or other data that must persist longer than the lifetime of a single EC2 instance, EBS volumes, Amazon EFS file systems, or Amazon S3 are more appropriate.	Amazon EC2 Amazon EBS Amazon EFS Amazon S3
Relational database storage	In most cases, relational databases require storage that persists beyond the lifetime of a single EC2 instance, making EBS volumes the natural choice.	Amazon EC2 Amazon EBS
Shared storage	Instance store volumes are dedicated to a single EC2 instance and can't be shared with other systems or users. If you need storage that can be detached from one instance and attached to a different instance, or if you need the ability to share data easily, Amazon EFS, Amazon S3, or Amazon EBS are better choices.	Amazon EFS Amazon S3 Amazon EBS
Snapshots	If you need the convenience, long-term durability, availability, and the ability to share point-in-time disk snapshots, EBS volumes are a better choice.	Amazon EBS

Performance

The instance store volumes that are not SSD-based in most EC2 instance families have performance characteristics similar to standard EBS volumes. Because the EC2 instance virtual machine and the local instance store volumes are located on the same physical server, interaction with this storage is very fast, particularly for sequential access. To increase aggregate IOPS, or to improve sequential disk throughput, multiple instance store volumes can be grouped together using RAID 0 (disk striping) software. Because the bandwidth of the disks is not limited by the network, aggregate sequential throughput for multiple instance volumes can be higher than for the same number of EBS volumes.

Durability and Availability

Amazon EC2 local instance store volumes are not intended to be used as durable disk storage. Unlike Amazon EBS volume data, data on instance store volumes persists only during the life of the associated EC2 instance. This functionality means that data on instance store volumes is persistent across orderly instance reboots, but if the EC2 instance is stopped and restarted, terminates, or fails, all data on the instance store volumes is lost.

You should not use local instance store volumes for any data that must persist over time, such as permanent file or database storage, without providing data persistence by replicating data or periodically copying data to durable storage such as Amazon EBS or Amazon S3.

Interfaces

There is no separate management API for EC2 instance store volumes. Instead, instance store volumes are specified using the block device mapping feature of the Amazon EC2 API and the AWS Management Console. You cannot create or destroy instance store volumes, but you can control whether or not they are exposed to the EC2 instance and what device name is mapped to for each volume.

There is also no separate data API for instance store volumes. Just like EBS volumes, instance store volumes present a block-device interface to the EC2 instance. To the EC2 instance, an instance store volume appears just like a local disk drive. To write to and read data from instance store volumes, you use the native file system I/O interfaces of your chosen operating system.

Cost Model

The cost of an EC2 instance includes any local instance store volumes, if the instance type provides them. Although there is no additional charge for data storage on local instance store volumes, note that data transferred to and from Amazon EC2 instance store volumes from other Availability Zones or outside of an Amazon EC2 Region can incur data transfer charges; additional charges apply for use of any persistent storage, such as Amazon S3, Amazon Glacier, Amazon EBS volumes, and Amazon EBS snapshots.

Amazon EFS

Amazon Elastic File System (Amazon EFS) delivers a simple, scalable, elastic, highly available, and highly durable network file system as a service to EC2 instances. It supports Network File System versions 4 (NFSv4) and 4.1 (NFSv4.1), which makes it easy to migrate enterprise applications to AWS or build new ones. You can create and configure file systems quickly and easily through a simple web services interface. You don't need to provision storage in advance and there is no minimum fee or setup cost—you simply pay for what you use. Amazon EFS is designed to provide a highly scalable network file system that can grow to petabytes, which allows massively parallel access from EC2 instances to your data within a Region. It is also highly available and highly durable because it stores data and metadata across multiple Availability Zones in a Region.

To understand Amazon EFS, it is best to examine the different components that allow EC2 instances access to EFS file systems. You can create one or more EFS file systems within an AWS Region. Each file system is accessed by EC2 instances via mount targets, which are created per Availability Zone. You create one mount target per Availability Zone in the VPC you create using Amazon Virtual Private Cloud. Traffic flow between Amazon EFS and EC2 instances is controlled using security groups associated with the EC2 instance and the EFS mount targets. Access to EFS file system objects (files and directories) is controlled using standard Unix-style read/write/execute permissions based on user and group IDs.

Usage Patterns

Amazon EFS is designed to meet the needs of multi-threaded applications and applications that concurrently access data from multiple EC2 instances and that require substantial levels of aggregate throughput and input/output operations per second (IOPS). Its distributed design enables high levels of availability, durability, and scalability, which results in a small latency overhead for each file operation. This makes Amazon EFS ideal for growing datasets consisting of larger files that need both high performance and multi-client access.

Amazon EFS supports highly parallelized workloads and is designed to meet the performance needs of big data and analytics, media processing, content management, web serving, and home directories.

Amazon EFS doesn't suit all storage situations. The following table presents some storage needs for which you should consider other AWS storage options.

Storage Need	Solution	AWS Services
Archival data	Data that requires encrypted archival storage with infrequent read access with a long recovery time objective (RTO) can be stored in Amazon Glacier more cost-effectively.	Amazon Glacier
Relational database storage	In most cases, relational databases require storage that is mounted, accessed, and locked by a single node (EC2 instance, etc.). When running relational databases on AWS, look at leveraging Amazon RDS or Amazon EC2 with Amazon EBS PIOPS volumes.	Amazon RDS Amazon EC2 Amazon EBS
Temporary storage	Consider using local instance store volumes for needs such as scratch disks, buffers, queues, and caches.	Amazon EC2 Local Instance Store

Performance

Amazon EFS file systems are distributed across an unconstrained number of storage servers, enabling file systems to grow elastically to petabyte-scale and allowing massively parallel access from EC2 instances within a Region. This distributed data storage design means that multi-threaded applications and applications that concurrently access data from multiple EC2 instances can drive substantial levels of aggregate throughput and IOPS.

There are two different performance modes available for Amazon EFS: General Purpose and Max I/O. General Purpose performance mode is the default mode and is appropriate for most file systems. However, if your overall Amazon EFS workload will exceed 7,000 file operations per second per file system, we recommend the file system use Max I/O performance mode. Max I/O performance mode is optimized for applications where tens, hundreds, or thousands of EC2 instances are accessing the file system. With this mode, file systems scale to higher levels of aggregate throughput and operations per second with a tradeoff of slightly higher latencies for file operations.

Durability and Availability

Amazon EFS is designed to be highly durable and highly available. Each Amazon EFS file system object (such as a directory, file, or link) is redundantly stored across multiple Availability Zones within a Region. Amazon EFS is designed to be as highly durable and available as Amazon S3.

Interfaces

Amazon offers a network protocol-based HTTP (RFC 2616) API for managing Amazon EFS, as well as supporting for EFS operations within the AWS SDKs and the AWS CLI. If you prefer to work with a graphical user interface, the AWS Management Console gives you all the capabilities of the API in a browser interface. EFS file systems use Network File System version 4 (NFSv4) and version 4.1 (NFSv4.1) for data access. We recommend using NFSv4.1 to take advantage of the performance benefits in the latest version, including scalability and parallelism.

Cost Model

Amazon EFS provides the capacity you need, when you need it, without having to provision storage in advance. It is also designed to be highly available and highly durable as each file system object (such as a directory, file, or link) is redundantly stored across multiple Availability Zones. This highly durable, highly available architecture is built into the pricing model, and you only pay for the amount of storage you put into your file system. As files are added, your EFS file system dynamically grows, and you only pay for the amount of storage you use. As files are removed, your EFS file system dynamically shrinks, and you stop paying for the data you deleted. There are no charges for bandwidth or requests, and there are no minimum commitments or up-front fees.

Amazon CloudFront

[Amazon CloudFront](#) is a content-delivery web service that speeds up the distribution of your website's dynamic, static, and streaming content by making it available from a global network of edge locations. When a user requests content that you're serving with Amazon CloudFront, the user is routed to the edge location that provides the lowest latency (time delay), so content is delivered with better performance than if the user had accessed the content from a data center farther away. If the content is already in the edge location with the lowest latency, Amazon CloudFront delivers it immediately. If the content is not currently in that edge location, Amazon CloudFront retrieves it from an Amazon S3 bucket or an HTTP server (for example, a web server) that you have identified as the source for the definitive version of your content. Amazon CloudFront caches content at edge locations for a period of time that you specify.

Amazon CloudFront supports all files that can be served over HTTP. These files include dynamic web pages, such as HTML or PHP pages, and any popular static files that are a part of your web application, such as website images, audio, video, media files or software downloads. For on-demand media files, you can also choose to stream your content using Real-Time Messaging Protocol (RTMP) delivery. Amazon CloudFront also supports delivery of live media over HTTP.

Amazon CloudFront is optimized to work with other Amazon web services, such as Amazon S3, Amazon EC2, Elastic Load Balancing, and Amazon Route 53.

Amazon CloudFront also works seamlessly with any non-AWS origin servers that store the original, definitive versions of your files.

Usage Patterns

CloudFront is ideal for distribution of frequently accessed static content that benefits from edge delivery, such as popular website images, videos, media files or software downloads. Amazon CloudFront can also be used to deliver dynamic web applications over HTTP. These applications can include static content, dynamic content, or a whole site with a mixture of the two. Amazon CloudFront is also commonly used to stream audio and video files to web browsers and mobile devices.

If you need to remove an object from Amazon CloudFront edge-server caches before it expires, you can either [invalidate](#) the object or use [object versioning](#) to serve a different version of the object that has a different name. Additionally, it might be better to serve infrequently accessed data directly from the origin server, avoiding the additional cost of origin fetches for data that is not likely to be reused at the edge; however, origin fetches to Amazon S3 are free.

Performance

Amazon CloudFront is designed for low-latency and high-bandwidth delivery of content. Amazon CloudFront speeds up the distribution of your content by routing end users to the edge location that can best serve each end user's request in a worldwide network of edge locations. Typically, requests are routed to the nearest Amazon CloudFront edge location in terms of latency. This approach dramatically reduces the number of networks that your users' requests must pass through and improves performance. Users get both lower latency—here latency is the time it takes to load the first byte of an object—and the higher sustained data transfer rates needed to deliver popular objects at scale.

Durability and Availability

Because a CDN is an edge cache, Amazon CloudFront does not provide durable storage. The origin server, such as Amazon S3 or a web server running on Amazon EC2, provides the durable file storage needed. Amazon

CloudFront provides high availability by using a distributed global network of edge locations. Origin requests from the edge locations to AWS origin servers (for example, Amazon EC2, Amazon S3, and so on) are carried over network paths that Amazon constantly monitors and optimizes for both availability and performance. This edge network provides increased reliability and availability because there is no longer a central point of failure. Copies of your files are now held in edge locations around the world.

Interfaces

You can manage and configure Amazon CloudFront in several ways. The AWS Management Console provides an easy way to manage Amazon CloudFront and supports all features of the Amazon CloudFront API. You can also use the Amazon CloudFront command line tools, the native REST API, or one of the supported SDKs.

There is no data API for Amazon CloudFront and no command to preload data. Instead, data is automatically pulled into Amazon CloudFront edge locations on the first access of an object from that location.

Clients access content from CloudFront edge locations either using HTTP or HTTPS from locations across the Internet; these protocols are configurable as part of a given CloudFront distribution.

Cost Model

With Amazon CloudFront, there are no long-term contracts or required minimum monthly commitments—you pay only for as much content as you actually deliver through the service. Amazon CloudFront has two pricing components: regional data transfer out (per GB) and requests (per 10,000). As part of the Free Usage Tier, new AWS customers don't get charged for 50 GB data transfer out and 2,000,000 HTTP and HTTPS requests each month for one year.

Note that if you use an AWS service as the origin (for example, Amazon S3, Amazon EC2, Elastic Load Balancing, or others), data transferred from the origin to edge locations (i.e., Amazon CloudFront "origin fetches") will be free of charge. For web distributions, data transfer out of Amazon CloudFront to your origin server will be billed at the "Regional Data Transfer Out of Origin" rates.

CloudFront provides three different price classes according to where your content needs to be distributed. If you don't need your content to be distributed globally, but only within certain locations such as the US and Europe, you can lower the prices you pay to deliver by choosing a price class that includes only these locations.

Conclusion

Cloud storage is a critical component of cloud computing because it holds the information used by applications. Big data analytics, data warehouses, Internet of Things, databases, and backup and archive applications all rely on some form of data storage architecture.

Cloud storage is typically more reliable, scalable, and secure than traditional on-premises storage systems. AWS offers a complete range of cloud storage services to support both application and archival compliance requirements. This whitepaper provides guidance for understanding the different storage services and features available in the AWS Cloud. Usage patterns, performance, durability and availability, scalability and elasticity, security, interface, and cost models are outlined and described for these cloud storage services. While this gives you a better understanding of the features and characteristics of these cloud services, it is crucial for you to understand your workloads and requirements then decide which storage service is best suited for your needs.

Simple Storage Service (S3)

Amazon S3 Storage Classes

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/storage-class-intro.html>

Amazon S3 offers a range of storage classes designed for different use cases. These include Amazon S3 Standard for general-purpose storage of frequently accessed data, Amazon S3 Standard - Infrequent Access for long-lived, but less frequently accessed data, and Amazon Glacier for long-term archive. Amazon S3 also offers configurable lifecycle policies for managing your data throughout its lifecycle. Once a policy is set, your data will automatically migrate to the most appropriate storage class without any changes to your application.

STANDARD – This storage class is ideal for performance-sensitive use cases and frequently accessed data. STANDARD is the default storage class; if you don't specify storage class at the time that you upload an object, Amazon S3 assumes the STANDARD storage class.

STANDARD_IA – This storage class (IA, for infrequent access) is optimized for long-lived and less frequently accessed data, for example backups and older data where access has diminished, but the use case still demands high performance.

Note There is a retrieval fee associated with STANDARD_IA objects which makes it most suitable for infrequently accessed data.

The STANDARD_IA storage class is suitable for larger objects greater than 128 Kilobytes that you want to keep for at least 30 days.

GLACIER – The GLACIER storage class is suitable for archiving data where data access is infrequent. Archived objects are not available for real-time access. You must first restore the objects before you can access them.

The GLACIER storage class uses the very low-cost Amazon Glacier storage service, but you still manage objects in this storage class through Amazon S3.

Note the following about the GLACIER storage class:

- You cannot specify GLACIER as the storage class at the time that you create an object. You create GLACIER objects by first uploading objects using STANDARD, RRS, or STANDARD_IA as the storage class. Then, you transition these objects to the GLACIER storage class using lifecycle management.
- You must first restore the GLACIER objects before you can access them (STANDARD, RRS, and STANDARD_IA objects are available for anytime access).

All the preceding storage classes are designed to sustain the concurrent loss of data in two facilities

Amazon S3 also offers the following storage class that enables you to save costs by maintaining fewer redundant copies of your data.

REDUCED_REDUNDANCY – The Reduced Redundancy Storage (RRS) storage class is designed for noncritical, reproducible data stored at lower levels of redundancy than the STANDARD storage class, which reduces storage costs.

The durability level (see the following table) corresponds to an average annual expected loss of 0.01% of objects. For example, if you store 10,000 objects using the RRS option, you can, on average, expect to incur an annual loss of a single object per year (0.01% of 10,000 objects).

Note

This annual loss represents an expected average and does not guarantee the loss of less than 0.01% of objects in a given year.

RRS provides a cost-effective, highly available solution for distributing or sharing content that is durably stored elsewhere, or for storing thumbnails, transcoded media, or other processed data that can be easily reproduced. If an RRS object is lost, Amazon S3 returns a 405 error on requests made to that object. Amazon S3 can send an event notification to alert a user or start a workflow when it detects that an RRS object is lost. To receive notifications, you need to add notification configuration to your bucket.

The following table summarizes the durability and availability offered by each of the storage classes.

Storage Class	Durability (designed for)	Availability (designed for)	Other Considerations		
STANDARD	99.999999999%	99.99%	None		
STANDARD_IA	99.999999999%	99.9%	There is a retrieval fee associated with STANDARD_IA objects which makes it most suitable for infrequently accessed data. For pricing information, see Amazon S3 Pricing .		
GLACIER	99.999999999%	99.99% (after you restore objects)	GLACIER objects are not available for real-time access. You must first restore archived objects before you can access them. For more information, see Restoring Archived Objects .		
RRS	99.99%	99.99%	None		

Amazon S3 Using Versioning

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html>

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures.

Working with Delete Markers

A delete marker is a placeholder (marker) for a versioned object that was named in a simple DELETE request. Because the object was in a versioning-enabled bucket, the object was not deleted. The delete marker, however, makes Amazon S3 behave as if it had been deleted.

Deleting a Delete Marker will restore the object.

Amazon S3 Object Lifecycle Management

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html>

Lifecycle configuration enables you to specify the lifecycle management of objects in a bucket. The configuration is a set of one or more rules, where each rule defines an action for Amazon S3 to apply to a group of objects. These actions can be classified as follows:

Transition actions – In which you define when objects transition to another storage class. For example, you may choose to transition objects to the STANDARD_IA (IA, for infrequent access) storage class 30 days after creation, or archive objects to the GLACIER storage class one year after creation.

Expiration actions – In which you specify when the objects expire. Then Amazon S3 deletes the expired objects on your behalf.

Amazon S3 Object Tagging

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/object-tagging.html>

Object tagging enables you to categorize storage. Each tag is a key-value pair.

You can add tags to new objects when you upload them or you can add them to existing objects. Note the following:

- You can associate up to 10 tags with an object. Tags associated with an object must have unique tag keys.
- A tag key can be up to 128 Unicode characters in length and tag values can be up to 256 Unicode characters in length.
- Key and values are case sensitive.

Amazon S3 Object Key and Metadata

URL Link: <http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingMetadata.html>

Each Amazon S3 object has data, a key, and metadata. Object key (or key name) uniquely identifies the object in a bucket. Object metadata is a set of name-value pairs. You can set object metadata at the time you upload it. After you upload the object, you cannot modify object metadata. The only way to modify object metadata is to make a copy of the object and set the metadata.

Object Key

When you create an object, you specify the key name, which uniquely identifies the object in the bucket. For example, in the Amazon S3 console (see AWS Management Console), when you highlight a bucket, a list of objects in your bucket appears. These names are the object keys. The name for a key is a sequence of Unicode characters whose UTF-8 encoding is at most 1024 bytes long.

Note that the Amazon S3 data model is a flat structure: you create a bucket, and the bucket stores objects. There is no hierarchy of subbuckets or subfolders; however, you can infer logical hierarchy using key name prefixes and delimiters as the Amazon S3 console does. The Amazon S3 console supports a concept of folders.

Suppose your bucket (*companybucket*) has four objects with the following object keys:

Development/Projects1.xls
Finance/statement1.pdf
Private/taxdocument.pdf
s3-dg.pdf

The console uses the key name prefixes (*Development/*, *Finance/*, and *Private/*) and delimiter (*/*) to present a folder structure.

The *s3-dg.pdf* key does not have a prefix, so its object appears directly at the root level of the bucket.

Note

Amazon S3 supports buckets and objects, there is no hierarchy in Amazon S3. However, the prefixes and delimiters in an object key name, enables the Amazon S3 console and the AWS SDKs to infer hierarchy and introduce concept of folders.

Cross-Region Replication

URL Link <http://docs.aws.amazon.com/AmazonS3/latest/dev/crr.html>

Cross-region replication is a bucket-level feature that enables automatic, asynchronous copying of objects across buckets in different AWS regions. To activate this feature, you add a `replication` configuration to your source bucket. In the configuration, you provide information such as the destination bucket where you want

objects replicated to. You can request Amazon S3 to replicate all or a subset of objects with specific key name prefixes.

The object replicas in the destination bucket are exact replicas of the objects in the source bucket. They have the same key names and the same metadata—for example, creation time, owner, user-defined metadata, version ID, ACL, and storage class (assuming you did not explicitly specify different storage class for object replicas in the replication configuration). Amazon S3 encrypts all data in transit across AWS regions using SSL. You can also optionally specify storage class to use when Amazon S3 creates object replicas (if you don't specify this Amazon S3 assume storage class of the source object).

Use-case Scenarios

You might configure cross-region replication on a bucket for various reasons, including these:

- Compliance requirements – Although, by default, Amazon S3 stores your data across multiple geographically distant Availability Zones, compliance requirements might dictate that you store data at even further distances. Cross-region replication allows you to replicate data between distant AWS regions to satisfy these compliance requirements.
- Minimize latency – Your customers are in two geographic locations. To minimize latency in accessing objects, you can maintain object copies in AWS regions that are geographically closer to your users.
- Operational reasons – You have compute clusters in two different regions that analyze the same set of objects. You might choose to maintain object copies in those regions.

Optionally, if you have cost considerations, you can direct Amazon S3 to use the STANDARD_IA storage class for object replicas. For more information about cost considerations, see [Amazon S3 Pricing](#).

Requirements

Requirements for cross-region replication:

- The source and destination buckets must be versioning-enabled. For more information about versioning, see [Using Versioning](#).
- The source and destination buckets must be in different AWS regions. For a list of AWS regions where you can create a bucket, see [Regions and Endpoints](#) in the AWS General Reference.
- You can replicate objects from a source bucket to only one destination bucket.
- Amazon S3 must have permission to replicate objects from that source bucket to the destination bucket on your behalf. You can grant these permissions by creating an IAM role that Amazon S3 can assume. You must grant this role permissions for Amazon S3 actions so that when Amazon S3 assumes this role, it can perform replication tasks. For more information about IAM roles, see [Create an IAM Role](#).
- If the source bucket owner also owns the object, the bucket owner has full permissions to replicate the object. If not, the source bucket owner must have permission for the Amazon S3 actions `s3:GetObjectVersion` and `s3:GetObjectVersionACL` to read the object and object ACL. For more information about Amazon S3 actions, see [Specifying Permissions in a Policy](#). For more information about resources and ownership, see [Amazon S3 Resources](#).
- If you are setting up cross-region replication in a cross-account scenario (where the source and destination buckets are owned by different AWS accounts), the source bucket owner must have permission to replicate objects in the destination bucket. The destination bucket owner needs to grant these permissions via a bucket policy. For an example, see [Walkthrough 2: Configure Cross-Region Replication Where Source and Destination Buckets Are Owned by Different AWS Accounts](#).

URL Link <http://docs.aws.amazon.com/AmazonS3/latest/dev/crr-what-is-isnot-replicated.html>

This section explains what Amazon S3 replicates and what it does not replicate after you add a replication configuration on a bucket.

What Is Replicated

Amazon S3 replicates the following:

- Any new objects created after you add a replication configuration, with exceptions described in the next section.
- Objects created with server-side encryption using the Amazon S3-managed encryption key. The replicated copy of the object is also encrypted using server-side encryption using the Amazon S3-managed encryption key.
- Amazon S3 replicates only objects in the source bucket for which the bucket owner has permission to read objects and read ACLs. For more information about resource ownership, see [About the Resource Owner](#).
- Any object ACL updates are replicated, although there can be some delay before Amazon S3 can bring the two in sync. This applies only to objects created after you add a replication configuration to the bucket.
- S3 replicates object tags, if any.

Delete Operation and Cross-Region Replication

If you delete an object from the source bucket, the cross-region replication behavior is as follows:

- If a DELETE request is made without specifying an object version ID, Amazon S3 adds a delete marker, which cross-region replication replicates to the destination bucket. For more information about versioning and delete markers, see [Using Versioning](#).
- If a DELETE request specifies a particular object version ID to delete, Amazon S3 deletes that object version in the source bucket, but it does not replicate the deletion in the destination bucket (in other words, it does not delete the same object version from the destination bucket). This behavior protects data from malicious deletions.

What Is Not Replicated

Amazon S3 does not replicate the following:

- Amazon S3 does not retroactively replicate objects that existed before you added replication configuration.
- Objects created with server-side encryption using either customer-provided (SSE-C) or AWS KMS-managed encryption (SSE-KMS) keys are not replicated. For more information about server-side encryption, see [Protecting Data Using Server-Side Encryption](#). Amazon S3 does not keep the encryption keys you provide after the object is created in the source bucket so it cannot decrypt the object for replication, and therefore it does not replicate the object.
- Amazon S3 does not replicate objects in the source bucket for which the bucket owner does not have permissions. If the object owner is different from the bucket owner, see [Granting Cross-Account Permissions to Upload Objects While Ensuring the Bucket Owner Has Full Control](#).
- Updates to bucket-level subresources are not replicated. This allows you to have different bucket configurations on the source and destination buckets. For more information about resources, see [Amazon S3 Resources](#).
- Only customer actions are replicated. Actions performed by lifecycle configuration are not replicated. For more information lifecycle configuration, see [Object Lifecycle Management](#). For example, if

lifecycle configuration is enabled only on your source bucket, Amazon S3 creates delete markers for expired objects, but it does not replicate those markers. However, you can have the same lifecycle configuration on both the source and destination buckets if you want the same lifecycle actions to happen to both buckets.

- Objects in the source bucket that are replicas, created by another cross-region replication, are not replicated. Suppose you configure cross-region replication where bucket A is the source and bucket B is the destination. Now suppose you add another cross-region replication where bucket B is the source and bucket C is the destination. In this case, objects in bucket B that are replicas of objects in bucket A will not be replicated to bucket C.

[Amazon S3 Transfer Acceleration](#)

URL Link <http://docs.aws.amazon.com/AmazonS3/latest/dev/transfer-acceleration.html>

Amazon S3 Transfer Acceleration enables fast, easy, and secure transfers of files over long distances between your client and an S3 bucket. Transfer Acceleration takes advantage of Amazon CloudFront's globally distributed edge locations. As the data arrives at an edge location, data is routed to Amazon S3 over an optimized network path.

When using Transfer Acceleration, additional data transfer charges may apply. For more information about pricing, see [Amazon S3 Pricing](#).

Why Use Amazon S3 Transfer Acceleration?

You might want to use Transfer Acceleration on a bucket for various reasons, including the following:

- You have customers that upload to a centralized bucket from all over the world.
- You transfer gigabytes to terabytes of data on a regular basis across continents.
- You underutilize the available bandwidth over the Internet when uploading to Amazon S3.

For more information about when to use Transfer Acceleration, see [Amazon S3 FAQs](#).

Using the Amazon S3 Transfer Acceleration Speed Comparison Tool

You can use the [Amazon S3 Transfer Acceleration Speed Comparison tool](#) to compare accelerated and non-accelerated upload speeds across Amazon S3 regions. The Speed Comparison tool uses multipart uploads to transfer a file from your browser to various Amazon S3 regions with and without using Transfer Acceleration.

You can access the Speed Comparison tool using either of the following methods:

- Copy the following URL into your browser window, replacing *region* with the region that you are using (for example, us-west-2) and *yourBucketName* with the name of the bucket that you want to evaluate: `http://s3-accelerate-speedtest.s3-accelerate.amazonaws.com/en/accelerate-speed-comparison.html?region=region&origBucketName=yourBucketName`
- For a list of the regions supported by Amazon S3, see [Regions and Endpoints](#) in the Amazon Web Services General Reference.
- Use the Amazon S3 console. For details, see [Enabling Transfer Acceleration](#) in the Amazon Simple Storage Service Console User Guide.

Getting Started with Amazon S3 Transfer Acceleration

To get started using Amazon S3 Transfer Acceleration perform the following steps:

1. **Enable Transfer Acceleration on a bucket** – For your bucket to work with transfer acceleration, the bucket name must conform to DNS naming requirements and must not contain periods ("."). You can enable Transfer Acceleration on a bucket any of the following ways:
 - Use the Amazon S3 console. For more information, see [Enabling Transfer Acceleration](#) in the Amazon Simple Storage Service Console User Guide.
 - Use the REST API [PUT Bucket accelerate](#) operation.
 - Use the AWS CLI and AWS SDKs. For more information, see [Using the AWS SDKs, CLI, and Explorers](#).
2. **Transfer data to and from the acceleration-enabled bucket by using one of the following s3-accelerate endpoint domain names:**
 - `bucketname.s3-accelerate.amazonaws.com` – to access an acceleration-enabled bucket.
 - `bucketname.s3-accelerate.dualstack.amazonaws.com` – to access an acceleration-enabled bucket over IPv6. Amazon S3 dual-stack endpoints support requests to S3 buckets over IPv6 and IPv4. The Transfer Acceleration dual-stack endpoint only uses the virtual hosted-style type of endpoint name. For more information, see [Getting Started Making Requests over IPv6](#) and [Using Amazon S3 Dual-Stack Endpoints](#).

Note You can continue to use the regular endpoint in addition to the accelerate endpoints.

After Transfer Acceleration is enabled, it can take up to 20 minutes for you to realize the performance benefit. However, the accelerate endpoint will be available as soon as you enable Transfer Acceleration.

Amazon Glacier

What Is Amazon Glacier?

URL Link <http://docs.aws.amazon.com/amazonglacier/latest/dev/introduction.html>

Welcome to the Amazon Glacier Developer Guide. Amazon Glacier is a storage service optimized for infrequently used data, or "cold data."

Amazon Glacier is an extremely low-cost storage service that provides durable storage with security features for data archiving and backup. With Amazon Glacier, customers can store their data cost effectively for months, years, or even decades. Amazon Glacier enables customers to offload the administrative burdens of operating and scaling storage to AWS, so they don't have to worry about capacity planning, hardware provisioning, data replication, hardware failure detection and recovery, or time-consuming hardware migrations. For more service highlights and pricing information, go to the [Amazon Glacier detail page](#).

Amazon Glacier is a great storage choice when low storage cost is paramount, your data is rarely retrieved, and retrieval latency of several hours is acceptable. If your application requires fast or frequent access to your data, consider using Amazon S3. For more information, go to [Amazon Simple Storage Service \(Amazon S3\)](#).

Are You a First-Time Amazon Glacier User?

If you are a first-time user of Amazon Glacier, we recommend that you begin by reading the following sections:

- What is Amazon Glacier—The rest of this section describes the underlying data model, the operations it supports, and the AWS SDKs that you can use to interact with the service.
- Getting Started—The [Getting Started with Amazon Glacier](#) section walks you through the process of creating a vault, uploading archives, creating jobs to download archives, retrieving the job output, and deleting archives.

Important Amazon Glacier provides a management console, which you can use to create and delete vaults. However, all other interactions with Amazon Glacier require that you use the AWS Command Line Interface (CLI) or write code. For example, to upload data, such as photos, videos, and other

documents, you must either use the AWS CLI or write code to make requests, by using either the REST API directly or by using the AWS SDKs. For more information about using Amazon Glacier with the AWS CLI, go to [AWS CLI Reference for Amazon Glacier](#). To install the AWS CLI, go to [AWS Command Line Interface](#).

HINT Glacier requires you to use the CLI or write code. You don't need to know how to develop for Glacier, just know that it can be done. We will be going to build scripting/coding skills in future lessons.

Elastic Compute Cloud (EC2)

Amazon EC2 Root Device Volume

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/RootDeviceStorage.html>

When you launch an instance, the root device volume contains the image used to boot the instance. When we introduced Amazon EC2, all AMIs were backed by Amazon EC2 instance store, which means the root device for an instance launched from the AMI is an instance store volume created from a template stored in Amazon S3. After we introduced Amazon EBS, we introduced AMIs that are backed by Amazon EBS. This means that the root device for an instance launched from the AMI is an Amazon EBS volume created from an Amazon EBS snapshot.

You can choose between AMIs backed by Amazon EC2 instance store and AMIs backed by Amazon EBS. We recommend that you use AMIs backed by Amazon EBS, because they launch faster and use persistent storage.

Root Device Storage Concepts

You can launch an instance from either an instance store-backed AMI or an Amazon EBS-backed AMI. The description of an AMI includes which type of AMI it is; you'll see the root device referred to in some places as either ebs (for Amazon EBS-backed) or instance store (for instance store-backed). This is important because there are significant differences between what you can do with each type of AMI.

Instance Store-backed Instances

Instances that use instance stores for the root device automatically have one or more instance store volumes available, with one volume serving as the root device volume. When an instance is launched, the image that is used to boot the instance is copied to the root volume. Note that you can optionally use additional instance store volumes, depending on the instance type.

Any data on the instance store volumes persists as long as the instance is running, but this data is deleted when the instance is terminated (instance store-backed instances do not support the Stop action) or if it fails (such as if an underlying drive has issues).

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

Amazon EBS-backed Instances

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

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

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

- Stop and then start again (try this method first).
- Automatically snapshot all relevant volumes and create a new AMI. For more information, see [Creating an Amazon EBS-Backed Linux AMI](#).
- Attach the volume to the new instance by following these steps:
 1. Create a snapshot of the root volume.
 2. Register a new AMI using the snapshot.
 3. Launch a new instance from the new AMI.
 4. Detach the remaining Amazon EBS volumes from the old instance.
 5. Reattach the Amazon EBS volumes to the new instance.

Amazon AMI

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html>

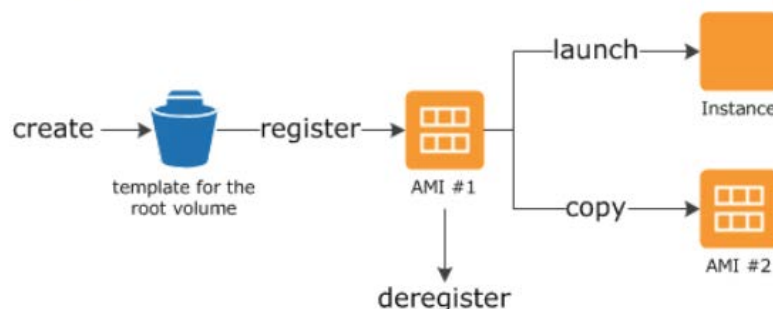
An Amazon Machine Image (AMI) provides the information required to launch an instance, which is a virtual server in the cloud. You specify an AMI when you launch an instance, and you can launch as many instances from the AMI as you need. You can also launch instances from as many different AMIs as you need.

An AMI includes the following:

- A template for the root volume for the instance (for example, an operating system, an application server, and applications)
- Launch permissions that control which AWS accounts can use the AMI to launch instances
- A block device mapping that specifies the volumes to attach to the instance when it's launched

Using an AMI

The following diagram summarizes the AMI lifecycle. After you create and register an AMI, you can use it to launch new instances. (You can also launch instances from an AMI if the AMI owner grants you launch permissions.) You can copy an AMI to the same region or to different regions. When you are finished launching instance from an AMI, you can deregister the AMI.



You can search for an AMI that meets the criteria for your instance. You can search for AMIs provided by AWS or AMIs provided by the community.

When you are connected to an instance, you can use it just like you use any other server.

Creating Your Own AMI

You can customize the instance that you launch from a public AMI and then save that configuration as a custom AMI for your own use. Instances that you launch from your AMI use all the customizations that you've made. The root storage device of the instance determines the process you follow to create an AMI. The root volume of an instance is either an Amazon EBS volume or an instance store volume.

To help categorize and manage your AMIs, you can assign custom tags to them.

Buying, Sharing, and Selling AMIs

After you create an AMI, you can keep it private so that only you can use it, or you can share it with a specified list of AWS accounts. You can also make your custom AMI public so that the community can use it. Building a safe, secure, usable AMI for public consumption is a fairly straightforward process, if you follow a few simple guidelines.

You can purchase an AMIs from a third party, including AMIs that come with service contracts from organizations such as Red Hat. You can also create an AMI and sell it to other Amazon EC2 users.

Deregistering Your AMI

You can deregister an AMI when you have finished with it. After you deregister an AMI, you can't use it to launch new instances.

Amazon Linux

The Amazon Linux AMI is a supported and maintained Linux image provided by AWS. The following are some of the features of Amazon Linux:

- A stable, secure, and high-performance execution environment for applications running on Amazon EC2.
- Provided at no additional charge to Amazon EC2 users.
- Repository access to multiple versions of MySQL, PostgreSQL, Python, Ruby, Tomcat, and many more common packages.
- Updated on a regular basis to include the latest components, and these updates are also made available in the yum repositories for installation on running instances.
- Includes packages that enable easy integration with AWS services, such as the AWS CLI, Amazon EC2 API and AMI tools, the Boto library for Python, and the Elastic Load Balancing tools.

Elastic Block Storage (EBS)

Amazon EBS Volumes

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumes.html>

An Amazon EBS volume is a durable, block-level storage device that you can attach to a single EC2 instance. EBS volumes persist independently from the running life of an EC2 instance. After a volume is attached to an instance, you can use it like any other physical hard drive. EBS volumes are flexible. You can dynamically grow volumes, modify provisioned IOPS capacity, and change volume types on live production volumes. Amazon EBS provides the following volume types: General Purpose SSD (gp2), Provisioned IOPS SSD (io1), 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.

Benefits of Using EBS Volumes

EBS volumes provide several.

Data availability When you create an EBS volume in an Availability Zone, it is automatically replicated within that zone to prevent data loss due to failure of any single hardware component. After you create a volume, you can attach it 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.

An EBS volume can be attached to only one instance at a time within the same Availability Zone. However, multiple volumes can be attached to a single instance. 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.

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.

By default, EBS volumes that are attached to a running instance automatically detach from the instance with their data intact when that instance is terminated. The volume can then be reattached to a new instance, enabling quick recovery. 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 data persists on the volume until the volume is deleted explicitly. By default, EBS volumes (root volumes) that are created and attached to an instance at launch are 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.

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 be attached to a running instance in order to take a snapshot. 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 restored from encrypted snapshots are automatically encrypted. By optionally specifying a different Availability Zone, you can use this functionality to create duplicate a volume in that zone.

Snapshots are incremental backups, meaning that only the blocks on the volume that have changed after your most recent snapshot are saved. 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 restore the volume. To help categorize and manage your volumes and snapshots, you can tag them with metadata of your choice.

Flexibility EBS volumes support live configuration changes while in production. You can modify volume type, volume size, and IOPS capacity without service interruptions.

Creating an Amazon EBS-Backed Linux AMI

URL Link <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/creating-an-ami-ebs.html>

To create an Amazon EBS-backed Linux AMI, start from an instance that you've launched from an existing Amazon EBS-backed Linux AMI. After you've customized the instance to suit your needs, create and register a new AMI, which you can use to launch new instances with these customizations.

The procedures described below will work for Amazon EC2 instances backed by encrypted Amazon EBS volumes (including the root volume) as well as for unencrypted volumes.

The AMI creation process is different for instance store-backed AMIs.

Overview of Creating Amazon EBS-Backed AMIs

First, launch an instance from an AMI that's similar to the AMI that you'd like to create. You can connect to your instance and customize it. When the instance is configured correctly, ensure data integrity by stopping the instance before you create an AMI, then create the image. When you create an Amazon EBS-backed AMI, we automatically register it for you.

Amazon EC2 powers down the instance before creating the AMI to ensure that everything on the instance is stopped and in a consistent state during the creation process. If you're confident that your instance is in a consistent state appropriate for AMI creation, you can tell Amazon EC2 not to power down and reboot the instance. Some file systems, such as XFS, can freeze and unfreeze activity, making it safe to create the image without rebooting the instance.

During the AMI-creation process, Amazon EC2 creates snapshots of your instance's root volume and any other EBS volumes attached to your instance. If any volumes attached to the instance are encrypted, the new AMI only launches successfully on instances that support Amazon EBS encryption.

Depending on the size of the volumes, it can take several minutes for the AMI-creation process to complete (sometimes up to 24 hours). You may find it more efficient to create snapshots of your volumes prior to creating your AMI. This way, only small, incremental snapshots need to be created when the AMI is created, and the process completes more quickly (the total time for snapshot creation remains the same).

After the process completes, you have a new AMI and snapshot created from the root volume of the instance. When you launch an instance using the new AMI, we create a new EBS volume for its root volume using the snapshot. Both the AMI and the snapshot incur charges to your account until you delete them.

If you add instance-store volumes or EBS volumes to your instance in addition to the root device volume, the block device mapping for the new AMI contains information for these volumes, and the block device mappings for instances that you launch from the new AMI automatically contain information for these volumes. The instance-store volumes specified in the block device mapping for the new instance are new and don't contain any data from the instance store volumes of the instance you used to create the AMI. The data on EBS volumes persists.

EBS Snapshot

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-creating-snapshot.html>

After writing data to an EBS volume, you can periodically create a snapshot of the volume to use as a baseline for new volumes or for data backup. If you make periodic snapshots of a volume, the snapshots are incremental so that only the blocks on the device that have changed after your last 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 restore the volume.

Snapshots occur asynchronously; the point-in-time snapshot is created immediately, but the status of the snapshot is pending until the snapshot is complete (when all of the modified blocks have been transferred to Amazon S3), which can take several hours for large initial snapshots or subsequent snapshots where many blocks have changed. While it is completing, an in-progress snapshot is not affected by ongoing reads and writes to the volume.

Important Although you can take a snapshot of a volume while a previous snapshot of that volume is in the *pending* status, having multiple *pending* snapshots of a volume may result in reduced volume performance until the snapshots complete.

There is a limit of 5 *pending* snapshots for a single *gp2*, *io1*, or Magnetic volume, and 1 *pending* snapshot for a single *st1* or *sc1* volume. If you receive a *ConcurrentSnapshotLimitExceeded* error while trying to create multiple concurrent snapshots of the same volume, wait for one or more of the *pending* snapshots to complete before creating another snapshot of that volume.

By default, only you can create volumes from snapshots that you own. However, you can share your unencrypted snapshots with specific AWS accounts, or you can share them with the entire AWS community by making them public.

You can take a snapshot of an attached volume that is in use. However, snapshots only capture data that has been written to your Amazon EBS volume at the time the snapshot command is issued. This might exclude any data that has been cached by any applications or the operating system. If you can pause any file writes to the volume long enough to take a snapshot, your snapshot should be complete. However, if you can't pause all file writes to the volume, you should unmount the volume from within the instance, issue the snapshot command, and then remount the volume to ensure a consistent and complete snapshot. You can remount and use your volume while the snapshot status is *pending*.

To create a snapshot for Amazon EBS volumes that serve as root devices, you should stop the instance before taking the snapshot.

To unmount the volume in Linux, use the following command:

```
umount -d device_name
```

Where *device_name* is the device name (for example, `/dev/sdh`).

After you've created a snapshot, you can tag it to help you manage it later. For example, you can add tags describing the original volume from which the snapshot was created, or the device name that was used to attach the original volume to an instance.

Restoring an Amazon EBS Volume from a Snapshot

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-restoring-volume.html>

You can restore an Amazon EBS volume with data from a snapshot stored in Amazon S3. You need to know the ID of the snapshot you wish to restore your volume from and you need to have access permissions for the snapshot.

New volumes created from existing EBS snapshots load lazily in the background. This means that after a volume is created from a snapshot, there is no need to wait for all of the data to transfer from Amazon S3 to your EBS volume before your attached instance can start accessing the volume and all its data. If your instance accesses data that hasn't yet been loaded, the volume immediately downloads the requested data from Amazon S3, and continues loading the rest of the data in the background.

RAID Configuration

URL Link: <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/raid-config.html>

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. For greater I/O performance than you can achieve with a single volume, RAID 0 can stripe multiple volumes together; for on-instance redundancy, RAID 1 can mirror two volumes together.

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.

Note

You should avoid booting from a RAID volume. Grub is typically installed on only one device in a RAID array, and if one of the mirrored devices fails, you may be unable to boot the operating system.

RAID Configuration Options

The following table compares the common RAID 0 and RAID 1 options.

Configuration	Use	Advantages	Disadvantages
RAID 0	When I/O performance is more important than fault tolerance; for example, as in a heavily used database (where data replication is already set up separately).	I/O is distributed across the volumes in a stripe. If you add a volume, you get the straight addition of throughput.	Performance of the stripe is limited to the worst performing volume in the set. Loss of a single volume results in a complete data loss for the array.
RAID 1	When fault tolerance is more important than I/O performance; for example, as in a critical application.	Safer from the standpoint of data durability.	Does not provide a write performance improvement; requires more Amazon EC2 to Amazon EBS bandwidth than non-RAID configurations because the data is written to multiple volumes simultaneously.

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.

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. A RAID 1 array offers a "mirror" of your data for extra redundancy. Before you perform this procedure, you need to decide how large your RAID array should be and how many IOPS you want to provision.

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. The resulting size and bandwidth of a RAID 1 array is equal to the size and bandwidth of the volumes in the array. For example, two 500 GiB Amazon EBS volumes with 4,000 provisioned IOPS each will create a 1000 GiB RAID 0 array with an available bandwidth of 8,000 IOPS and 640 MB/s of throughput or a 500 GiB RAID 1 array with an available bandwidth of 4,000 IOPS and 320 MB/s of throughput.

This documentation provides basic RAID setup examples. For more information about RAID configuration, performance, and recovery, see the Linux RAID Wiki at https://raid.wiki.kernel.org/index.php/Linux_Raid.

Amazon EC2 Instance Store

What is Amazon EC2 Instance Store

URL Link <http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html>

An **instance store** provides temporary block-level storage for your instance. This storage is located on disks that are physically attached to the host computer. Instance store is ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers.

An instance store consists of one or more instance store volumes exposed as block devices. The size of an instance store as well as the number of devices available varies by instance type. While an instance store is dedicated to a particular instance, the disk subsystem is shared among instances on a host computer.

The virtual devices for instance store volumes are `ephemeral[0-23]`. Instance types that support one instance store volume have `ephemeral0`. Instance types that support two instance store volumes have `ephemeral0` and `ephemeral1`, and so on.

Instance Store Lifetime

You can specify instance store volumes for an instance only when you launch it. You can't detach an instance store volume from one instance and attach it to a different instance.

The data in an instance store persists only during the lifetime of its associated instance. If an instance reboots (intentionally or unintentionally), data in the instance store persists. However, data in the instance store is lost under the following circumstances:

- The underlying disk drive fails
- The instance stops
- The instance terminates

Therefore, do not rely on instance store for valuable, long-term data. Instead, use more durable data storage, such as Amazon S3, Amazon EBS, or Amazon EFS.

If you create an AMI from an instance, the data on its instance store volumes isn't preserved and isn't present on the instance store volumes of the instances that you launch from the AMI.

Instance Store Volumes

The instance type determines the size of the instance store available and the type of hardware used for the instance store volumes. Instance store volumes are included as part of the instance's hourly cost. You must specify the instance store volumes that you'd like to use when you launch the instance (except for NVMe instance store volumes, which are available by default), and then format and mount them before using them. You can't make an instance store volume available after you launch the instance. For more information, see [Add Instance Store Volumes to Your EC2 Instance](#).

Some instance types use NVMe or SATA based solid state drives (SSD) to deliver very high random I/O performance. This is a good option when you need storage with very low latency, but you don't need the data to persist when the instance terminates or you can take advantage of fault-tolerant architectures. For more information, see [SSD Instance Store Volumes](#).

The following table provides the quantity, size, type, and performance optimizations of instance store volumes available on each supported instance type. For a complete list of instance types, including EBS-only types, see [Amazon EC2 Instance Types](#).

Instance Type	Instance Store Volumes	Type	Needs Initialization*	TRIM Support**
c1.medium	1 x 350 GB†	HDD	✓	
c1.xlarge	4 x 420 GB (1,680 GB)	HDD	✓	
c3.large	2 x 16 GB (32 GB)	SSD	✓	
c3.xlarge	2 x 40 GB (80 GB)	SSD	✓	
c3.2xlarge	2 x 80 GB (160 GB)	SSD	✓	
c3.4xlarge	2 x 160 GB (320 GB)	SSD	✓	
c3.8xlarge	2 x 320 GB (640 GB)	SSD	✓	
cc2.8xlarge	4 x 840 GB (3,360 GB)	HDD	✓	
cg1.4xlarge	2 x 840 GB (1,680 GB)	HDD	✓	
cr1.8xlarge	2 x 120 GB (240 GB)	SSD	✓	
d2.xlarge	3 x 2,000 GB (6 TB)	HDD		
d2.2xlarge	6 x 2,000 GB (12 TB)	HDD		
d2.4xlarge	12 x 2,000 GB (24 TB)	HDD		
d2.8xlarge	24 x 2,000 GB (48 TB)	HDD		
g2.2xlarge	1 x 60 GB	SSD	✓	
g2.8xlarge	2 x 120 GB (240 GB)	SSD	✓	
hi1.4xlarge	2 x 1,024 GB (2,048 GB)	SSD		
hs1.8xlarge	24 x 2,000 GB (48 TB)	HDD	✓	
i2.xlarge	1 x 800 GB	SSD		✓
i2.2xlarge	2 x 800 GB (1,600 GB)	SSD		✓
i2.4xlarge	4 x 800 GB (3,200 GB)	SSD		✓
i2.8xlarge	8 x 800 GB (6,400 GB)	SSD		✓
i3.large	1 x 475 GB	NVMe SSD		✓
i3.xlarge	1 x 950 GB	NVMe SSD		✓
i3.2xlarge	1 x 1,900 GB	NVMe SSD		✓
i3.4xlarge	2 x 1,900 GB (3.8 TB)	NVMe SSD		✓
i3.8xlarge	4 x 1,900 GB (7.6 TB)	NVMe SSD		✓
i3.16xlarge	8 x 1,900 GB (15.2 TB)	NVMe SSD		✓
m1.small	1 x 160 GB†	HDD	✓	
m1.medium	1 x 410 GB	HDD	✓	
m1.large	2 x 420 GB (840 GB)	HDD	✓	
m1.xlarge	4 x 420 GB (1,680 GB)	HDD	✓	
m2.xlarge	1 x 420 GB	HDD	✓	
m2.2xlarge	1 x 850 GB	HDD	✓	
m2.4xlarge	2 x 840 GB (1,680 GB)	HDD	✓	
m3.medium	1 x 4 GB	SSD	✓	
m3.large	1 x 32 GB	SSD	✓	
m3.xlarge	2 x 40 GB (80 GB)	SSD	✓	
m3.2xlarge	2 x 80 GB (160 GB)	SSD	✓	
r3.large	1 x 32 GB	SSD		✓
r3.xlarge	1 x 80 GB	SSD		✓
r3.2xlarge	1 x 160 GB	SSD		✓
r3.4xlarge	1 x 320 GB	SSD		✓
r3.8xlarge	2 x 320 GB (640 GB)	SSD		✓
x1.16xlarge	1 x 1,920 GB	SSD		
x1.32xlarge	2 x 1,920 GB (3,840 GB)	SSD		

Add Instance Store Volumes to Your EC2 Instance

URL Link <http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/add-instance-store-volumes.html>

You specify the EBS volumes and instance store volumes for your instance using a block device mapping. Each entry in a block device mapping includes a device name and the volume that it maps to. The default block device mapping is specified by the AMI you use.

A block device mapping always specifies the root volume for the instance. The root volume is either an Amazon EBS volume or an instance store volume. For more information, see [Storage for the Root Device](#). The root volume is mounted automatically. For instances with an instance store volume for the root volume, the size of this volume varies by AMI, but the maximum size is 10 GB.

You can use a block device mapping to specify additional EBS volumes when you launch your instance, or you can attach additional EBS volumes after your instance is running. For more information, see [Amazon EBS Volumes](#).

You can specify the instance store volumes for your instance only when you launch an instance. You can't attach instance store volumes to an instance after you've launched it.

The number and size of available instance store volumes for your instance varies by instance type. Some instance types do not support instance store volumes. For more information about the instance store volumes support by each instance type, see [Instance Store Volumes](#). If the instance type you choose for your instance supports instance store volumes, you must add them to the block device mapping for the instance when you launch it. After you launch the instance, you must ensure that the instance store volumes for your instance are formatted and mounted before you can use them. Note that the root volume of an instance store-backed instance is mounted automatically.

Adding Instance Store Volumes to an AMI

You can create an AMI with a block device mapping that includes instance store volumes. After you add instance store volumes to an AMI, any instance that you launch from the AMI includes these instance store volumes. Note that when you launch an instance, you can omit volumes specified in the AMI block device mapping and add new volumes.

Adding Instance Store Volumes to an Instance

When you launch an instance, the default block device mapping is provided by the specified AMI. If you need additional instance store volumes, you must add them to the instance as you launch it. Note that you can also omit devices specified in the AMI block device mapping.

Making Instance Store Volumes Available on Your Instance

After you launch an instance, the instance store volumes are available to the instance, but you can't access them until they are mounted. For Linux instances, the instance type determines which instance store volumes are mounted for you and which are available for you to mount yourself. For Windows instances, the EC2Config service mounts the instance store volumes for an instance. 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 recommends.

Many instance store volumes are pre-formatted with the ext3 file system. SSD-based instance store volumes that support TRIM instruction are not pre-formatted with any file system. However, you can format volumes with the file system of your choice after you launch your instance. For more information, see [Instance Store](#)

Volume TRIM Support. For Windows instances, the EC2Config service reformats the instance store volumes with the NTFS file system.

You can confirm that the instance store devices are available from within the instance itself using instance metadata. For more information, see [Viewing the Instance Block Device Mapping for Instance Store Volumes](#).

For Windows instances, you can also view the instance store volumes using Windows Disk Management. For more information, see [Listing the Disks Using Windows Disk Management](#).

Elastic File System (EFS)

What Is Amazon Elastic File System?

URL Link <http://docs.aws.amazon.com/efs/latest/ug/whatisefs.html>

Amazon Elastic File System (Amazon EFS) provides simple, scalable file storage for use with Amazon EC2. With Amazon EFS, storage capacity is elastic, growing and shrinking automatically as you add and remove files, so your applications have the storage they need, when they need it.

Amazon EFS has a simple web services interface that allows you to create and configure file systems quickly and easily. The service manages all the file storage infrastructure for you, avoiding the complexity of deploying, patching, and maintaining complex file system deployments.

Amazon EFS supports the Network File System version 4.1 (NFSv4.1) protocol, so the applications and tools that you use today work seamlessly with Amazon EFS. Multiple Amazon EC2 instances can access an Amazon EFS file system at the same time, providing a common data source for workloads and applications running on more than one instance or server.

With Amazon EFS, you pay only for the storage used by your file system. You don't need to provision storage in advance and there is no minimum fee or setup cost. For more information, see [Amazon EFS Pricing](#).

The service is designed to be highly scalable, highly available, and highly durable. Amazon EFS file systems store data and metadata across multiple Availability Zones in a region and can grow to petabyte scale, drive high levels of throughput, and allow massively parallel access from Amazon EC2 instances to your data.

Amazon EFS provides file system access semantics, such as strong data consistency and file locking. For more information, see [Data Consistency in Amazon EFS](#). Amazon EFS also allows you to tightly control access to your file systems through POSIX permissions.

Amazon EFS is designed to provide the throughput, IOPS, and low latency needed for a broad range of workloads. With Amazon EFS, throughput and IOPS scale as a file system grows, and file operations are delivered with consistent, low latencies. For more information, see [Amazon EFS Performance](#).

Note

Using Amazon EFS with Microsoft Windows Amazon EC2 instances is not supported.

Amazon EFS: How It Works

URL Link <http://docs.aws.amazon.com/efs/latest/ug/how-it-works.html>

Following, you can find a description about how Amazon EFS works, its implementation details, and security considerations.

Overview

Amazon EFS provides file storage in the AWS Cloud. With Amazon EFS, you can create a file system, mount the file system on an Amazon EC2 instance, and then read and write data from to and from your file system. You can mount an Amazon EFS file system in your VPC, through the Network File System version 4.1 (NFSv4.1) protocol.

For a list of Amazon EC2 Linux Amazon Machine Images (AMIs) that support this protocol, see [NFS Support](#). We recommend using a current generation Linux NFSv4.1 client, such as those found in Amazon Linux and Ubuntu AMIs. For some AMIs, you'll need to install an NFS client to mount your file system on your Amazon EC2 instance. For instructions, see [Installing the NFS Client](#).

You can access your Amazon EFS file system concurrently from Amazon EC2 instances in your Amazon VPC, so applications that scale beyond a single connection can access a file system. Amazon EC2 instances running in multiple Availability Zones within the same region can access the file system, so that many users can access and share a common data source.

Note the following restrictions:

- You can mount an Amazon EFS file system on instances in only one VPC at a time.
- Both the file system and VPC must be in the same AWS Region.

For a list of AWS regions where you can create an Amazon EFS file system, see the [Amazon Web Services General Reference](#).

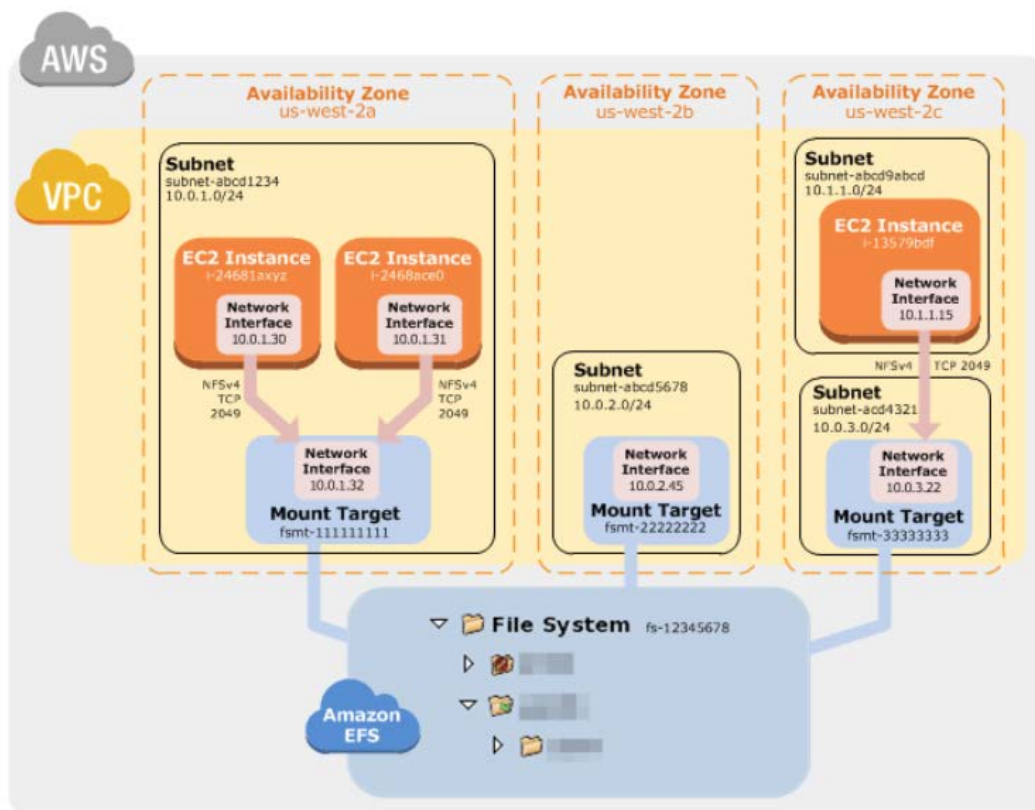
To access your Amazon EFS file system in a VPC, you create one or more mount targets in the VPC. You mount your file system from these mount targets. A mount target represents an IP address for an NFSv4.1 endpoint at which you can mount an Amazon EFS file system. You can create one mount target in each of the Availability Zones. If there are multiple subnets in an Availability Zone in your VPC, you create a mount target in one of the subnets, and all EC2 instances in that Availability Zone share that mount target.

Mount targets themselves are designed to be highly available. When designing your application for high availability and the ability to failover to other Availability Zones, keep in mind that the IP addresses and DNS for your mount targets in each Availability Zone are static.

After mounting the file system via the mount target, you use it like any other POSIX-compliant file system. For information about NFS-level permissions and related considerations, see [Network File System \(NFS\)–Level Users, Groups, and Permissions](#).

How Amazon EFS Works with Amazon EC2

The following illustration shows an example VPC accessing an Amazon EFS file system. Here, EC2 instances in the VPC have file systems mounted.



In this illustration, the VPC has three Availability Zones, and each has one mount target created in it. We recommend that you access the file system from a mount target within the same Availability Zone. Note that one of the Availability Zones has two subnets. However, a mount target is created in only one of the subnets. Creating this setup works as follows:

- 1 Create your Amazon EC2 resources and launch your Amazon EC2 instance. For more information on Amazon EC2, see [Amazon EC2 - Virtual Server Hosting](#).
- 2 Create your Amazon EFS file system.
- 3 Connect to your Amazon EC2 instance, and mount the Amazon EFS file system.

For detailed steps, see [Getting Started with Amazon Elastic File System](#).

Implementation Summary

In Amazon EFS, a file system is the primary resource. Each file system has properties such as ID, creation token, creation time, file system size in bytes, number of mount targets created for the file system, and the file system state. For more information, see [CreateFileSystem](#).

Amazon EFS also supports other resources to configure the primary resource. These include mount targets and tags:

- **Mount target** – To access your file system, you must create mount targets in your VPC. Each mount target has the following properties: the mount target ID, the subnet ID in which it is created, the file system ID for which it is created, an IP address at which the file system may be mounted, and the mount target state. You can use the IP address or the DNS name in your `mount` command. Each file system has a DNS name of the following form.

file-system-id.efs.aws-region.amazonaws.com

You can specify this DNS name in your `mount` command to mount the Amazon EFS file system. Suppose you create an `~/efs-mount-point` subdirectory on your EC2 instance or on-premises server. Then, you can use the `mount` command to mount the file system. For example, on an Amazon Linux AMI, you can use the following `mount` command.

```
$ sudo mount -t nfs4 -o
nfsvers=4.1,rsize=1048576,wsize=1048576,hard,timeo=600,retrans=2 file-system-DNS-
name:/ ~/efs-mount-point
```

For more information, see [Creating Mount Targets](#). First, you need to install the NFS client on your EC2 instance. The [Getting Started](#) exercise provides step-by-step instructions.

- **Tags** – To help organize your file systems, you can assign your own metadata to each of the file systems you create. Each tag is a key-value pair.

You can think of mount targets and tags as subresources that don't exist without being associated with a file system.

Authentication and Access Control

You must have valid credentials to make Amazon EFS API requests, such as create a file system. In addition, you must also have permissions to create or access resources. By default, when you use the root account credentials of your AWS account you can create and access resources owned by that account. However, we do not recommend using root account credentials. In addition, any AWS Identity and Access Management (IAM) users and roles you create in your account must be granted permissions to create or access resources. For more information about permissions, see [Authentication and Access Control for Amazon EFS](#).

[Getting Started with Amazon Elastic File System](#)

URL Link <http://docs.aws.amazon.com/efs/latest/ug/getting-started.html>

This Getting Started exercise shows you how to quickly create an Amazon Elastic File System (Amazon EFS) file system, mount it on an Amazon Elastic Compute Cloud (Amazon EC2) instance in your VPC, and test the end-to-end setup.

There are four steps you need to perform to create and use your first Amazon EFS file system:

- Create your Amazon EC2 resources and launch your instance.
- Create your Amazon EFS file system.
- Connect to your Amazon EC2 instance and mount the Amazon EFS file system.
- Clean up your resources and protect your AWS account.

Assumptions

For this exercise, we assume the following:

- You're already familiar with using the Amazon EC2 console to launch instances.
- Your Amazon VPC, Amazon EC2, and Amazon EFS resources are all in the same region. This guide uses the US West (Oregon) Region (us-west-2).
- You have a default VPC in the region that you're using for this Getting Started exercise. If you don't have a default VPC, or if you want to mount your file system from a new VPC with new or existing security groups, you can still use this Getting Started exercise as long as you configure [Security Groups for EC2 Instances and Mount Targets](#).
- You have not changed the default inbound access rule for the default security group.

You can use the root credentials of your AWS account to sign in to the console and try the Getting Started exercise. However, AWS Identity and Access Management (IAM) recommends that you do not use the root credentials of your AWS account. Instead, create an administrator user in your account and use those credentials to manage resources in your account. For more information, see [Setting Up](#).

Relational Database Service (RDS)

Working With Backups

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_WorkingWithAutomatedBackups.html

Amazon RDS creates and saves automated backups of your DB instance. Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases.

Amazon RDS creates automated backups of your DB instance during the backup window of your DB instance. Amazon RDS saves the automated backups of your DB instance according to the backup retention period that you specify. If necessary, you can recover your database to any point in time during the backup retention period.

Your DB instance must be in the `ACTIVE` state for automated backups to occur. If your database is in another state, for example `STORAGE_FULL`, automated backups do not occur.

You can also backup your DB instance manually, by manually creating a DB snapshot.

Backup Storage

Your Amazon RDS backup storage for each region is composed of the automated backups and manual DB snapshots for that region. Your backup storage is equivalent to the sum of the database storage for all instances in that region. Moving a DB snapshot to another region increases the backup storage in the destination region.

All automated backups are deleted when you delete a DB instance. After you delete a DB instance, the automated backups can't be recovered. If you choose to have Amazon RDS create a final DB snapshot before it deletes your DB instance, you can use that to recover your DB instance.

Manual snapshots are not deleted.

The Backup Window

Automated backups occur daily during the preferred backup window. If the backup requires more time than allotted to the backup window, the backup continues after the window ends, until it finishes. The backup window can't overlap with the weekly maintenance window for the DB instance.

During the automatic backup window, storage I/O might be suspended briefly while the backup process initializes (typically under a few seconds). You may experience elevated latencies for a few minutes during backups for Multi-AZ deployments. For MariaDB, MySQL, Oracle, and PostgreSQL, I/O activity is not suspended on your primary during backup for Multi-AZ deployments, because the backup is taken from the standby. For SQL Server, I/O activity is suspended briefly during backup for Multi-AZ deployments.

If you don't specify a preferred backup window when you create the DB instance, Amazon RDS assigns a default 30-minute backup window which is selected at random from an 8-hour block of time per region.

The Backup Retention Period

You can set the backup retention period when you create a DB instance. If you don't set the backup retention period, the default backup retention period is one day if you create the DB instance using the Amazon RDS API or the AWS CLI, or seven days if you create the DB instance using the AWS Console. For Amazon Aurora DB

clusters, the default backup retention period is one day regardless of how the DB cluster is created. After you create a DB instance, you can modify the backup retention period. You can set the backup retention period to between 1 and 35 days. You can also set the backup retention period to 0, which disables automated backups. Manual snapshot limits (50 per region) do not apply to automated backups.

Important An outage occurs if you change the backup retention period from 0 to a non-zero value or from a non-zero value to 0.

Disabling Automated Backups

You may want to temporarily disable automated backups in certain situations; for example, while loading large amounts of data.

Important We highly discourage disabling automated backups because it disables point-in-time recovery. Disabling automatic backups for a DB instance deletes all existing automated backups for the instance. If you disable and then re-enable automated backups, you are only able to restore starting from the time you re-enabled automated backups.

Restoring From a DB Snapshot

URL Link: http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_RestoreFromSnapshot.html

Amazon RDS creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. You can create a DB instance by restoring from this DB snapshot. When you restore the DB instance, you provide the name of the DB snapshot to restore from, and then provide a name for the new DB instance that is created from the restore. You cannot restore from a DB snapshot to an existing DB instance; a new DB instance is created when you restore.

You can change to a different edition of the DB engine when restoring from a DB snapshot only if the DB snapshot has the required storage allocated for the new edition. For example, to change from SQL Server Web Edition to SQL Server Standard Edition, the DB snapshot must have been created from a SQL Server DB instance that had at least 200 GB of allocated storage, which is the minimum allocated storage for SQL Server Standard edition.

You can restore a DB instance and use a different storage type than the source DB snapshot. In this case the restoration process will be slower because of the additional work required to migrate the data to the new storage type. In the case of restoring to or from **Magnetic (Standard)** storage, the migration process is the slowest as **Magnetic** storage does not have the IOPS capability of **Provisioned IOPS** or **General Purpose (SSD)** storage.

Note Amazon RDS does not support changing the storage configuration for a SQL Server DB instance when restoring from a DB snapshot.

Hint Restoring a DB Instance creates a new database with a new endpoint.

Restoring a DB Instance to a Specified Time

URL Link http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIT.html

The Amazon RDS automated backup feature automatically creates a storage volume snapshot of your DB instance, backing up the entire DB instance and not just individual databases. This backup occurs during a daily user-configurable 30 minute period known as the backup window. Automated backups are kept for a configurable number of days (called the backup retention period). You can restore your DB instance to any specific time during this retention period, creating a new DB instance.

When you restore a DB instance to a point in time, the default DB security group is applied to the new DB instance. If you need custom DB security groups applied to your DB instance, you must apply them explicitly

using the AWS Management Console, the Amazon RDS API **ModifyDBInstance** action, or the AWS CLI **modify-db-instance** command once the DB instance is available.

You can restore to any point in time during your backup retention period. To determine the latest restorable time for a DB instance, use the AWS CLI [describe-db-instances](#) command and look at the value returned in the **LatestRestorableTime** field for the DB instance. The latest restorable time for a DB instance is typically within 5 minutes of the current time.

The OFFLINE, EMERGENCY, and SINGLE_USER modes are not currently supported. Setting any database into one of these modes will cause the latest restorable time to stop moving ahead for the whole instance.

Several of the database engines used by Amazon RDS have special considerations when restoring from a point in time. When you restore an Oracle DB instance to a point in time, you can specify a different Oracle DB engine, license model, and DBName (SID) to be used by the new DB instance. When you restore a SQL Server DB instance to a point in time, each database within that instance is restored to a point in time within 1 second of each other database within the instance. Transactions that span multiple databases within the instance may be restored inconsistently.

Some actions, such as changing the recovery model of a SQL Server database, can break the sequence of logs that are used for point-in-time recovery. In some cases, Amazon RDS can detect this issue and the latest restorable time is prevented from moving forward; in other cases, such as when a SQL Server database uses the BULK_LOGGED recovery model, the break in log sequence is not detected. It may not be possible to restore a SQL Server DB instance to a point in time if there is a break in the log sequence. For these reasons, Amazon RDS does not support changing the recovery model of SQL Server databases.

Copying a DB Snapshot or DB Cluster Snapshot

URL Link http://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_CopySnapshot.html

With Amazon Relational Database Service (Amazon RDS), you can copy DB snapshots and DB cluster snapshots in the following ways:

- Copy an automated or manual DB snapshot or DB cluster snapshot to create a manual snapshot in the same AWS Region.
- Within the same AWS account, copy an automated or manual DB snapshot or DB cluster snapshot within a region, or from one region to another region.
- Within the same region, copy an automated or manual DB cluster snapshot from one AWS account to another AWS account.

You can't copy a DB cluster snapshot between regions and AWS accounts in a single step, but instead must perform one step for each of these copy actions.

As an alternative to copying, you can also share manual snapshots with other AWS accounts.

You can copy snapshots shared to you by other AWS accounts

Depending on the regions involved and the amount of data to be copied, a cross-region snapshot copy can take hours to complete. If there are large numbers of cross-region snapshot copy requests from a given source region, Amazon RDS might queue new cross-region copy requests for that source region until some in-progress copies have completed. No progress information is displayed about copy requests while they are in the queue. Progress information is displayed when the copy starts.

Working with Snapshot Retention

Amazon RDS deletes automated snapshots at the end of their retention period, when you disable automated snapshots for a DB instance or DB cluster, or when you delete a DB instance or DB cluster. If you want to keep an automated snapshot for a longer period, copy it to create a manual snapshot, which is retained until you delete it. Amazon RDS storage costs might apply to manual snapshots if they exceed your default storage space.

Limitations

There are some limitations to how and where you can copy snapshots:

- You can't copy a snapshot to or from the AWS GovCloud (US) region.
- You can't copy a DB snapshot across regions if it was created from a DB instance that is using Oracle Transparent Data Encryption (TDE) or Microsoft SQL Server TDE.
- You can't copy a SQL Server DB snapshot across regions if the DB snapshot was created from an instance using Multi-AZ mirroring.

Copying a DB Snapshot

For each AWS account, you can copy up to five DB snapshots at a time from one region to another. If you copy a DB snapshot to another AWS Region, you create a manual DB snapshot that is retained in that region. Copying a DB snapshot out of the source region incurs Amazon RDS data transfer charges.

To copy a DB snapshot, use the AWS Management Console, the [copy-db-snapshot](#) command, or the [CopyDBSnapshot](#) API action.

If you copy a DB snapshot using the `copy-db-snapshot` AWS CLI command or `CopyDBSnapshot` RDS API action, issue the command in the AWS region that you want to copy the DB snapshot to, and use an Amazon RDS Amazon Resource Name (ARN) to specify the source DB snapshot to be copied.

After the DB snapshot copy has been created in the new region, the DB snapshot copy behaves the same as all other DB snapshots in that region.

You can copy both encrypted and unencrypted DB snapshots for the following database engines:

- MariaDB
- MySQL
- Oracle
- PostgreSQL
- Microsoft SQL Server

WordPress Web Server

Uploading Files

URL Link https://codex.wordpress.org/Uploading_Files

About Uploading Files with WordPress

To upload files, you can use WordPress's online interface, the **Dashboard** or [one of the recommended editors](#) and upload your files via FTP.

This article tells you how to upload files using the **Dashboard**. To upload files via FTP, read [Uploading WordPress to a Remote Host](#).

About Uploading Files on Dashboard

After you log in to WordPress and click on the **Dashboard** menu at the top of the screen, you can upload files with the Flash uploader.

Dashboard lets you upload files in the following ways:

- [Immediately in a post.](#)
- [Immediately in a page.](#)
- [For later use in your Media Library.](#)

WordPress supports uploading the following file types:

Images

- .jpg
- .jpeg
- .png
- .gif
- .ico

Documents

- .pdf (Portable Document Format; Adobe Acrobat)
- .doc, .docx (Microsoft Word Document)
- .ppt, .pptx, .pps, .ppsx (Microsoft PowerPoint Presentation)
- .odt (OpenDocument Text Document)
- .xls, .xlsx (Microsoft Excel Document)
- .psd (Adobe Photoshop Document)

Audio

- .mp3
- .m4a
- .ogg
- .wav

Video

- .mp4, .m4v (MPEG-4)
- .mov (QuickTime)
- .wmv (Windows Media Video)
- .avi
- .mpg
- .ogv (Ogg)
- .3gp (3GPP)
- .3g2 (3GPP2)

Not all webhosts permit these files to be uploaded. Also, they may not permit large file uploads. If you are having issues, please check with your host first.

[Configure http for URL Re-Write](#)

URL Re-Write is done by configuring the httpd dameon conf file to allow OverRide. In simple terms, we can store content, such as images, in a different location than on the Web Server, and users browsing our website will be redirected. We will use in our lab to re-direct traffic to CloudFront CDN for image content.

Lesson Summary

Simple Storage Service (S3)

Simple Storage Service (S3) Overview

S3 Storage Class / Tiers

- S3 (durable, immediately available, frequently accessed)
- S3 - IA (durable, immediately available, infrequently accessed)
- S3 - Reduced Redundancy Storage (Data is easily reproducible)
- Glacier - Archived Data (Wait 3-5 hours before accessing)

- S3 - 99.99% Availability, 99.999999999% durability, sustain loss of 2 facilities
- S3-IA - 99.99% Availability, 99.999999999% durability, sustain loss of 2 facilities
- S3-RR - 99.99% Availability, 99.99% durability, sustain loss of 1 facility
- Glacier - Cheap. \$0.01 per gigabyte/month

S3 Core Fundamentals

- Key (name)
- Value (data)
- Version ID
- Metadata
- Sub-resources
- Access Control Lists

S3 Basics

- Amazon guarantees 99.99% availability for the S3 platform
- Amazon guarantees 99.999999999% durability for S3 data (11 x 9's)
- Tiered Storage Available
- Lifecycle Management
- Versioning
- Encryption
- Secure your data using access Control Lists and Bucket Policies

S3 Charges

- Storage
- Requests
- Data Transfer Pricing

- Object based storage only (for files)
- Not suitable to install an operating system

S3 stores data in alphabetical order (lexicographical order)

S3 - Versioning

- Stores all versions of an object
 - (Including all writes and/or delete an object)
- Great Backup Tool
- Once enabled, can only be suspended.
 - You cannot disable
- Integrates with Lifecycle Rules

S3 – Lifecycle

Can be used in conjunction with versioning

Can be applied to current and previous version

Following actions can be done:

- Transition to the Standard - Infrequent Access Storage Class

 - (128kb and 30 days after creation date)

- Archive to the Glacier Storage Class

 - (30 days after IA (60 total), if relevant)

 - (1 day of not using IA)

- Permanently Delete

 - (1 day after Glacier)

S3 - Cross Region Replication

Versioning must be enabled on both the source and destination buckets

Regions must be unique

Files in an existing bucket are not replicated automatically. All subsequent updated files will be replicated automatically

You cannot replicate to multiple buckets or use daisy chaining (at this time)

Delete markers are replicated

Deleting individual version or delete markers will not be replicated

Understand what Cross Region Replication is at a high level

Great way to backup/protect data

Versioning must be enabled to enable Cross Region Replication

AWS Create Message:

- Existing objects will not be replicated. Cross-Region Replication replicates every future upload of every object to another bucket.

S3 - Transfer Acceleration

S3 Transfer Acceleration utilizes CloudFront edge Network to accelerate uploads to S3 buckets.

- Use distinct URL to upload directly to edge location, which then transfers file to S3

 - (ex. `awslabweb.s3.accelerate.amazonaws.com`)

Bucket is hosted in Region 1

- Users are around the world

- Instead of users sending via internet connection, it is routed to nearest edge location.

- AWS has optimized their edge backbone to make it faster

Enable Transfer Acceleration

- Message about data transfers being 300% faster

 - Charges only if there is performance improvement

- Visit Speed Comparison Tool

 - Comparison tool will show comparison of Region/AZ around the world

This is a good option where majority of users are far away.

Little or worse performance if majority of users are in same Region/AZ

Amazon Glacier

Amazon Glacier Overview

Amazon Glacier is an extremely low-cost storage service that provides durable storage with security features for data archiving and backup. With Amazon Glacier, customers can store their data cost effectively for months, years, or even decades.

Amazon Glacier is a great storage choice when low storage cost is paramount, your data is rarely retrieved, and retrieval latency of several hours is acceptable. If your application requires fast or frequent access to your data, consider using Amazon S3

Important Amazon Glacier provides a management console, which you can use to create and delete vaults. However, all other interactions with Amazon Glacier require that you use the AWS Command Line Interface (CLI) or write code.

Alternatively, you can leverage Glacier through S3 Console using Bucket Lifecycle configuration rules

EBS - Elastic Block Storage

Volumes and Snapshots

Volumes vs Snapshots

- Volumes exist on EBS

 - Virtual hard disk

- Snapshots exist on S3

- Taking a snapshot of a volume, will store the snapshot on S3

- Snapshots are point in time copies of Volumes

- Snapshots are incremental, only the blocks that have changed since last snapshot, will be moved to S3

- First snapshot may take some time to create

Snapshots of Root Device Volumes

To create a snapshot for Amazon EBS root volumes, you should stop the instance before taking the snapshot

Amazon Machine Images (AMI)

AMIs are regional

You can only launch an AMI from the region in which it is stored

However, you can copy AMIs to other regions using the console, CLI or EC2 API

AWS does not copy launch permissions, user-defined tags, or Amazon S3 bucket permissions from the source AMI to the new AMI

EBS and Instance Store Volumes

There are two types of volumes

- Root volume (where Operating System is installed)

- Additional volumes (this is your data drives, example D:\ or /dev/sdb, etc)

Root Volume Sizes

- Root device volumes can be either EBS volumes or Instance Store volumes

 - An instance store root device volume's maximum size is 10 Gb

 - EBS root device volume can be up to 1 or 2 Tb depending on the OS

EBS Consists of:

General Purpose SSD: GP2 (Up to 10,000 IOPS)
Provisioned IOPS SSD: IO1 (More than 10,000 IOPS)
Magnetic: Cheap, infrequently accessed storage
Throughput Optimized HDD (st1)
Cold HDD (sc1)

You cannot mount 1 EBS volume to multiple EC2 instances, use EFS instead

EBS Backed vs Instance Store

Think hard disk

EBS backed volumes are persistent

EBS Volumes can be stopped, data will persist

EBS Volumes can be detached and reattached to other EC2 instances

By default, both Root volumes will be deleted on termination, however with EBS volume, you can keep the root device by unselecting the "Delete on Termination" option when creating the instance.

Other EBS volumes attached to the instance are preserved however, if you delete the instance

Instance Store backed volumes are not persistent (ephemeral)

Instance Store Volumes cannot be stopped. If you do this data will be wiped

If the underlying host fails, data is lost

Instance store volumes cannot be detached and reattached to other instances. They exist only for the life of that instance

Instance store device root volumes are terminated by DEFAULT when the EC2 instance is terminated. You CANNOT stop this

Other instance store volumes will be deleted on termination automatically

Other EBS volumes attached to the EC2 instance will persist automatically

You can reboot both, and will not lose your data

EBS Backed = Store Data Long Term

Instance Store = shouldn't be used for long term data storage.

RAID Array

How to take application consistent snapshot

Stop the application from writing to disk

Flush all caches to disk

How?

Freeze the file system

Unmount the RAID Array

Shut down the associate EC2 Instances

Elastic File System (EFS)

EFS Features

Supports the Network File System version 4 (NFSv4) protocol

You only pay for the storage you use (no pre-provisioning required)

Can scale up to petabytes

Can support thousands of concurrent NFS connections

Data is stored across multiple Availability Zones within a region
Read After Write Consistency

Relational Database Service (RDS)

Database Backups and Snapshots

Two Types of Backups

- Automated Backups
- Database Snapshots

Automated Backups

Automated Backups allow you to recover your database to any point in time within a retention period
Retention period can be between 1 and 35 days (default = 7 days)
Automated backups will take a full daily snapshot and will also store transaction logs throughout the day
When doing a recovery, AWS will first choose most recent daily backup, and then apply transaction logs relevant to that day
This allows for point in time recovery down to a second, within the retention period

Automated Backups are enabled by default

The backup data is stored in S3 and you get free storage equal to the size of your database

Example, an RDS instance of 10GB will get 10GB of storage for backup

Backups are taken within a defined window

During backup window, storage I/O may be suspended while your data is being backed up, and you may experience elevated latency

Database Snapshots

Db Snapshots are done manually (i.e. they are user initiated)

They are stored even after you delete the original RDS instance, unlike automated backups

Restoring Backups

Whenever you restore either an Automatic Backup or a manual Snapshot, the restored version of the database will be a new RDS instance with a new end point