AWS Storage Services Overview

A Look at Storage Services Offered by AWS

December 2016

Contents

Abstract

Amazon Web Services (AWS) is a flexible, cost-effective, easy-to-use cloud computing platform. This whitepaper is designed to help architects and developers understand the different storage services and features available in the AWS Cloud. We provide an overview of each storage service or feature and describe usage patterns, performance, durability and availability, scalability and elasticity, security, interfaces, and the cost model.

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).

Amazon S3

Amazon Simple Storage Service (Amazon S3) provides developers and IT teams secure, durable, highly scalable object storage at a very low cost.[1] 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 zero 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

There are four common usage patterns for Amazon S3.

First, Amazon S3 is used to store and distribute static web content and media. This content can be delivered directly from Amazon S3 because each object in Amazon S3 has a unique HTTP URL. Alternatively, Amazon S3 can serve as an origin store for a content delivery network (CDN), such as Amazon CloudFront. The elasticity of Amazon S3 makes it particularly well suited for hosting web content that requires bandwidth for addressing extreme demand spikes. Also, because no storage provisioning is required, Amazon S3 works well for fast-growing websites hosting data-intensive, user-generated content, such as video- and photo-sharing sites.

Second, Amazon S3 is used to host entire static websites. Amazon S3 provides a low-cost, highly available, and highly scalable solution, including storage for static HTML files, images, videos, and client-side scripts in formats such as JavaScript.

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. Because of the horizontal scalability of Amazon S3, you can access your data from multiple computing nodes concurrently without being constrained by a single connection.

Finally, Amazon S3 is often used as a highly durable, scalable, and secure solution for backup and archiving of critical data. You can easily move cold data to Amazon Glacier using lifecycle management rules on data stored in Amazon S3. You can also use Amazon S3 cross-region replication to automatically copy objects across S3 buckets in different AWS Regions asynchronously, providing disaster recovery solutions for business continuity.[2]

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. If you access Amazon S3 using multiple threads, multiple applications, or multiple clients concurrently, total Amazon S3 aggregate throughput typically scales to rates that far exceed what any single server can generate or consume.

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.[3] 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 search engine such as Amazon CloudSearch or a database such as Amazon DynamoDB or Amazon RDS. In these scenarios, Amazon S3 stores the actual information, and the search engine or 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.

Amazon S3 Transfer Acceleration enables fast, easy, and secure transfer of files over long distances between your client and your Amazon S3 bucket. It leverages Amazon CloudFront globally distributed edge locations to route traffic to your Amazon S3 bucket over an Amazon-optimized network path. To get started with Amazon S3 Transfer Acceleration you first must enable it on an Amazon S3 bucket. Then modify your Amazon S3 PUT and GET requests to use the s3-accelerate endpoint domain name (<bucketname>.s3-accelerate.amazonaws.com). The Amazon S3 bucket can still be accessed using the regular endpoint. Some customers have measured performance improvements in excess of 500 percent when performing intercontinental uploads.

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 99.999999999 percent (11 nines) durability per object and 99.99 percent availability over a one-year period.

Additionally, you have a choice of enabling cross-region replication on each Amazon S3 bucket. Once enabled, cross-region replication automatically copies objects across buckets in different AWS Regions asynchronously, providing 11 nines of durability and 4 nines of availability on both the source and destination Amazon S3 objects.

Scalability and Elasticity

Amazon S3 has been designed to offer a very high level of automatic scalability and elasticity. Unlike a typical file system that encounters issues when storing a large number of files in a directory, Amazon S3 supports a virtually unlimited number of files in any bucket. Also, unlike a disk drive that has a limit on the total amount of data that can be stored before you must partition the data across drives and/or servers, an Amazon S3 bucket can store a virtually unlimited number of bytes. You can store any number of objects (files) in a single bucket, and Amazon S3 will automatically manage scaling and distributing redundant copies of your information to other servers in other locations in the same Region, all using Amazon's high-performance infrastructure.

Security

Amazon S3 is highly secure. It provides multiple mechanisms for fine-grained control of access to Amazon S3 resources, and it supports encryption.

You can manage access to Amazon S3 by granting other AWS accounts and users permission to perform the resource operations by writing an access policy.[4]

You can protect Amazon S3 data at rest by using server-side encryption,[5] in which you request Amazon S3 to encrypt your object before it's written to disks in data centers and decrypt it when you download the object or by using client-side encryption,[6] in which you encrypt your data on the client side and upload the encrypted data to Amazon S3. You can protect the data in transit by using Secure Sockets Layer (SSL) or client-side encryption.

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. Additionally, you can add an optional layer of security by enabling Multi-Factor Authentication (MFA) Delete for a bucket.[7] With this option enabled for a bucket, two forms of authentication are required to change the versioning state of the bucket or to permanently delete an object version: valid AWS account credentials plus a six-digit code (a single-use, time-based password) from a physical or virtual token device.

To track requests for access to your bucket, you can enable access logging.[8] Each access log record provides details about a single access request, such as the requester, bucket name, request time, request action, response status, and error code, if any. Access log information can be useful in security and access audits. It can also help you learn about your customer base and understand your Amazon S3 bill.

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.

Additionally, you can use the Amazon S3 notification feature to receive notifications when certain events happen in your bucket. Currently, Amazon S3 can publish events when an object is uploaded or when an object is deleted.

Notifications can be issued to Amazon Simple Notification Service (SNS) topics,[9] Amazon Simple Queue Service (SQS) queues,[10] and AWS Lambda functions.[11]

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.[12] You can find pricing information at the Amazon S3 pricing page.[13] There are Data Transfer IN and OUT fees if you enable Amazon S3 Transfer Acceleration on a bucket and the transfer performance is faster than regular Amazon S3 transfer. If we determine that Transfer Acceleration is not likely to be faster than a regular Amazon S3 transfer of the same object to the same destination, we will not charge for that use of Transfer Acceleration for that transfer, and may bypass the Transfer Acceleration system for that upload.

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.[14] With Amazon Glacier, you can reliably store your data for as little as $0.007 per gigabyte per month. Amazon Glacier enables you to offload the administrative burdens of operating and scaling storage to AWS so that you don’t have to worry about capacity planning, hardware provisioning, data replication, hardware failure detection and repair, or time-consuming hardware migrations.

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).[15] 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.[16] 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 99.999999999 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.

Scalability and Elasticity

Amazon Glacier scales to meet growing and often unpredictable storage requirements. A single archive is limited to 40 TB in size, but there is no limit to the total amount of data you can store in the service. Whether you're storing petabytes or gigabytes, Amazon Glacier automatically scales your storage up or down as needed.

Security

By default, only you can access your Amazon Glacier data. If other people need to access your data, you can set up data access control in Amazon Glacier by using the AWS Identity and Access Management (IAM) service.[17] To do so, simply create an IAM policy that specifies which account users have rights to operations on a given vault.

Amazon Glacier uses server-side encryption to encrypt all data at rest. Amazon Glacier handles key management and key protection for you by using one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256). Customers who want to manage their own keys can encrypt data prior to uploading it.

Amazon Glacier allows you to lock vaults where long-term records retention is mandated by regulations or compliance rules. You can set compliance controls on individual Amazon Glacier vaults and enforce these by using lockable policies.

For example, you might specify controls such as “undeletable records” or “time-based data retention” in a Vault Lock policy and then lock the policy from future edits. After it’s locked, the policy becomes immutable, and Amazon Glacier enforces the prescribed controls to help achieve your compliance objectives.

To help monitor data access, Amazon Glacier is integrated with AWS CloudTrail, allowing any API calls made to Amazon Glacier in your AWS account to be captured and stored in log files that are delivered to an Amazon S3 bucket that you specify.[18]

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. You can then use the Amazon Glacier API actions to upload and retrieve archives, to monitor the status of your jobs, and also to configure your vault to send you a notification through Amazon SNS when a job is complete.

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. You can specify an absolute or relative time period (including 0 days) after which the specified Amazon S3 objects should be transitioned to Amazon Glacier. The Amazon S3 API includes a RESTORE operation. The retrieval process from Amazon Glacier using RESTORE takes three to five hours, the same as other Amazon Glacier retrievals.

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. For more information on how to use Amazon Glacier from Amazon S3, see the Object Lifecycle Management section of the Amazon S3 Developer Guide.[19]

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. You can find pricing information at the Amazon Glacier pricing page.[20]

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.[21] 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. We recommend clients run NFSv4.1 to take advantage of the many performance benefits found in the latest version, including scalability and parallelism. 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. You can find more information about how EFS works in the *Amazon EFS User Guide*.[22]

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. Because of this per-operation overhead, overall throughput generally increases as the average input/output (I/O) size increases since the overhead is amortized over a larger amount of data. 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

Storage Need	Solution	AWS Services
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 files 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.

Due to the spiky nature of file-based workloads, Amazon EFS is optimized to burst at high-throughput levels for short periods of time, while delivering low levels of throughput the rest of the time. A credit system determines when an Amazon EFS file system can burst. Over time, each file system earns burst credits at a baseline rate, determined by the size of the file system, and uses these credits whenever it reads or writes data. A file system can drive throughput continuously at its baseline rate. It accumulates credits during periods of inactivity or when throughput is below its baseline rate. These accumulated burst credits allow a file system to drive throughput above its baseline rate. The file system can continue to drive throughput above its baseline rate as long as it has a positive burst credit balance. You can see the burst credit balance for a file system by viewing the BurstCreditBalance metric in Amazon CloudWatch.[23]

Newly created file systems start with a credit balance of 2.1 TiB with a baseline rate of 50 MiB/s per TiB of storage and a burst rate of 100 MiB/s.

The following list describes some examples of bursting behaviors for file systems of different sizes.

File system size (GiB)	Baseline aggregate throughput (MiB/s)	Burst aggregate throughput (MiB/s)	Maximum burst duration (hours)	% of time file system can burst
10	0.5	100	6.0	0.5%
256	12.5	100	6.9	12.5%
512	25.0	100	8.0	25.0%
1024	50.0	100	12.0	50.0%
1536	75.0	150	12.0	50.0%
2048	100.0	200	12.0	50.0%
3072	150.0	300	12.0	50.0%
4096	200.0	400	12.0	50.0%

Here are a few recommendations to get the most performance out of your Amazon EFS file system. Because of the distributed architecture of Amazon EFS, larger I/O workloads generally experience higher throughput. EFS file systems can be mounted by thousands of EC2 instances concurrently. If your application is parallelizable across multiple instances, you can drive higher throughput levels on your file system in aggregate across instances. If your application can handle asynchronous writes to your file system, and you're able to trade off consistency for speed, enabling asynchronous writes may improve performance. We recommend Linux kernel version 4 or later and NFSv4.1 for all clients accessing EFS file systems. When mounting EFS file systems, use the mount options recommended in the Mounting File Systems and Additional Mounting Considerations sections of the *Amazon EFS User Guide*.[24] [25]

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.

Scalability and Elasticity

Amazon EFS automatically scales your file system storage capacity up or down as you add or remove files without disrupting your applications, giving you just the storage you need, when you need it, and while eliminating time-consuming administration tasks associated with traditional storage management (such as planning, buying, provisioning, and monitoring). Your EFS file system can grow from an empty file system to multiple petabytes automatically, and there is no provisioning, allocating, or administration.

Security

There are three levels of access control to consider when planning your EFS file system security: IAM permissions for API calls; security groups for EC2 instances and mount targets; and Network File System-level users, groups, and permissions.

IAM enables access control for administering EFS file systems, allowing you to specify an IAM identity (either an IAM user or IAM role) so you can create, delete, and describe EFS file system resources. The primary resource in Amazon EFS is a file system. All other EFS resources, such as mount targets and tags, are referred to as subresources. Identity-based policies, like IAM policies, are used to assign permissions to IAM identities to manage the EFS resources and subresources. Amazon groups play a critical role in establishing network connectivity between EC2 instances and EFS file systems. You associate one security group with an EC2 instance and another security group with an EFS mount target associated with the file system. These security groups act as firewalls and enforce rules that define the traffic flow between EC2 instances and EFS file systems.

EFS file system objects work in a Unix-style mode, which defines permissions needed to perform actions on objects. Users and groups are mapped to numeric identifiers, which are mapped to EFS users to represent file ownership. Files and directories within Amazon EFS are owned by a single owner and a single group. Amazon EFS uses these numeric IDs to check permissions when a user attempts to access a file system object.

For more information about Amazon EFS security, see the *Amazon EFS User Guide*.[26]

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. The API actions and EFS operations are used to create, delete, and describe file systems; create, delete, and describe mount targets; create, delete, and describe tags; and describe and modify mount target security groups. 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.

You can find pricing information for Amazon EFS at the Amazon EFS pricing page.[27]

Amazon EBS

Amazon Elastic Block Store (Amazon EBS) volumes provide durable block-level storage for use with EC2 instances.[28] 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. To learn more about Amazon EBS durability, see the EBS Durability and Availability section of this whitepaper. 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.

You can find information about Amazon EBS previous generation Magnetic volumes at the Amazon EBS Previous Generation Volumes page.[29]

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 99.999999999 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. The following table presents the storage characteristics of the current generation volume types.

	SSD-Backed Provisioned IOPS (io1)	SSD-Backed General Purpose (gp2)*	HDD-Backed Throughput Optimized (st1)	HDD-Backed Cold (sc1)
Use Cases	I/O-intensive NoSQL and relational databases	Boot volumes, low-latency interactive apps, dev & test	Big data, data warehouse, log processing	Colder data requiring fewer scans per day
Volume Size	4 GiB – 16 TiB	1 GiB – 16 TiB	500 GiB – 16 TiB	500 GiB – 16 TiB
Max IOPS per Volume**	20,000	10,000	500	250
Max Throughput per Volume	320 MiB/s	160 MiB/s	500 MiB/s	250 MiB/s
Max IOPS per Instance	65,000	65,000	65,000	65,000
Max Throughput per Instance	1,250 MiB/s	1,250 MiB/s	1,250 MiB/s	1,250 MiB/s
Dominant Performance Attribute	IOPS	IOPS	MiB/s	MiB/s

*Default volume type

**io1/gp2 based on 16 KiB I/O; st1/sc1 based on 1 MiB I/O

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. These volumes have a throughput limit range of 128 MiB/second for volumes less than or equal to 170 GiB; for volumes over 170 GiB, this limit increases at the rate of 768 KiB/second per GiB to a maximum of 160 MiB/second (at 214 GiB and larger). You can see the percentage of I/O credits remaining in the burst buckets for gp2 volumes by viewing the BurstBalance metric in Amazon CloudWatch.[30]

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. You specify an IOPS rate when creating a volume, and then Amazon EBS delivers within 10 percent of the provisioned IOPS performance 99.9 percent of the time over a given year, when attached to an EBS-optimized instance. The io1 volumes can range in size from 4 GiB to 16 TiB, and you can provision up to 20,000 IOPS per volume. The ratio of IOPS provisioned to the volume size requested can be a maximum of 50. For example, a volume with 5,000 IOPS must be at least 100 GB in size.

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. These volumes deliver performance in terms of throughput, measured in MiB/s, and include the ability to burst up to 250MiB/s per TiB, with a baseline throughput of 40MiB/s per TiB and a maximum throughput of 500MiB/s per volume. The st1 volumes are designed to deliver the expected throughput performance 99 percent of the time and has enough I/O credits to support a full-volume scan at the burst rate. The st1 volumes can’t be used as boot volumes. You can see the throughput credits remaining in the burst bucket for st1 volumes by viewing the BurstBalance metric in Amazon CloudWatch.[31]

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). Similarly to st1, sc1 volumes provide a burst model and can burst up to 80 MiB/s per TiB, with a baseline throughput of 12 MiB/s per TiB and a maximum throughput of 250 MB/s per volume. The sc1 volumes are designed to deliver the expected throughput performance 99 percent of the time and have enough I/O credits to support a full-volume scan at the burst rate. The sc1 volumes can’t be used as boot volumes. You can see the throughput credits remaining in the burst bucket for sc1 volumes by viewing the BurstBalance metric in CloudWatch.[32]

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. When attached to EBS-optimized instances, provisioned IOPS volumes are designed to deliver within 10 percent of the provisioned IOPS performance 99.9 percent of the time within a given year. Newly created EBS volumes receive their maximum performance the moment they are available, and they don't require initialization (formerly known as prewarming). However, you must initialize the storage blocks on volumes that were restored from snapshots before you can access the block.[33]

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. For example, Cassandra using General Purpose (SSD) volumes for data but Throughput Optimized (HDD) volumes for logs, or Hadoop using General Purpose (SSD) volumes for both data and logs. 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. EBS volumes are designed for an annual failure rate (AFR) of between 0.1 and 0.2 percent, where failure refers to a complete or partial loss of the volume, depending on the size and performance of the volume. This means, if you have 1,000 EBS volumes over the course of a year, you can expect unrecoverable failures with 1 or 2 of your volumes. This AFR makes EBS volumes 20 times more reliable than typical commodity disk drives, which fail with an AFR of around 4 percent. Despite these very low EBS AFR numbers, we still recommend that you create snapshots of your EBS volumes to improve the durability of your data. The Amazon EBS snapshot feature makes it easy to take application-consistent backups of your data. For more information on EBS durability, see the *Amazon EBS Availability and Durability* section of the *Amazon EBS Product Details* page.[34]

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.) All EBS volume types offer durable snapshot capabilities and are designed for 99.999 percent availability. If your EBS volume does fail, all snapshots of that volume remain intact, and you can recreate your volume from the last snapshot point. 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.

Scalability and Elasticity

Using the AWS Management Console or the Amazon EBS API, you can easily and rapidly provision and release EBS volumes to scale in and out with your total storage demands.

The simplest approach is to create and attach a new EBS volume and begin using it together with your existing ones. However, if you need to expand the size of a single EBS volume, you can effectively resize a volume using a snapshot:

1. Detach the original EBS volume.
2. Create a snapshot of the original EBS volume's data in Amazon S3.

3. Create a new EBS volume from the snapshot but specify a larger size than the original volume.
4. Attach the new, larger volume to your EC2 instance in place of the original. (In many cases, an OS-level utility must also be used to expand the file system.)
5. Delete the original EBS volume.

Security

IAM enables access control for your EBS volumes, allowing you to specify who can access which EBS volumes.

EBS encryption enables data-at-rest and data-in-motion security. It offers seamless encryption of both EBS boot volumes and data volumes as well as snapshots, eliminating the need to build and manage a secure key management infrastructure. These encryption keys are Amazon-managed or keys that you create and manage using the AWS Key Management Service (AWS KMS).[35] Data-in-motion security occurs on the servers that host EC2 instances, providing encryption of data as it moves between EC2 instances and EBS volumes.

Access control plus encryption offers a strong defense-in-depth security strategy for your data. For more information, see Amazon EBS Encryption in the *Amazon EBS User Guide*.[36]

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. The API actions and EBS operations are used to create, delete, describe, attach, and detach EBS volumes for your EC2 instances; to create, delete, and describe snapshots from Amazon EBS to Amazon S3; and to copy snapshots from one region to another. 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.

You can find pricing information for Amazon EBS at the Amazon EBS pricing page.[37]

Amazon EC2 Instance Storage

Amazon EC2 instance store volumes (also called ephemeral drives) provide temporary block-level storage for many EC2 instance types.[38] 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. You can choose to expose the instance store volumes at instance launch time by specifying a block device mapping. For more information, see Block Device Mapping in the *Amazon EC2 User Guide*.[39]

AWS offers two EC2 instance families that are purposely built for storage-centric workloads. Performance specifications of the storage-optimized (i2) and dense-storage (d2) instance families are outlined in the following table.

	SSD-Backed Storage-Optimized (i2)	**HDD-Backed Dense-Storage (d2)**
Use Cases	NoSQL databases, like Cassandra and MongoDB, scale out transactional databases, data warehousing, Hadoop, and cluster file systems.	Massively Parallel Processing (MPP) data warehousing, MapReduce and Hadoop distributed computing, distributed file systems, network file systems, log or data-processing applications
Read Performance	365,000 Random IOPS	3.5 GiB/s*
Write Performance	315,000 Random IOPS	3.1 GiB/s*
Instance Store Max. Capacity	6.4 TiB SSD	48 TiB HDD
Optimized For	Very high random IOPS	High disk throughput
* 2MiB block size		

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. EC2 instance storage is well-suited for this purpose. It consists of the virtual machine's boot device (for instance store AMIs only), plus one or more additional volumes that are dedicated to the EC2 instance (for both Amazon EBS AMIs and instance store AMIs). 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. Example applications include NoSQL databases like Cassandra and MongoDB, clustered databases, and online transaction processing (OLTP) systems. 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. Example applications include data warehouses, Hadoop/MapReduce storage nodes, and parallel file systems.

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

Storage Need	Solution	AWS Services
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.

Because of the way that EC2 virtualizes disks, the first write operation to any location on an instance store volume performs more slowly than subsequent writes. For most applications, amortizing this cost over the lifetime of the instance is acceptable. However, if you require high disk performance, we recommend that you prewarm your drives by writing once to every drive location before production use. The i2, r3, and hi1 instance types use direct-attached SSD backing that provides maximum performance at launch time without prewarming.

Additionally, r3 and i2 instance store-backed volumes support the TRIM command on Linux instances. For these volumes, you can use TRIM to notify the SSD controller whenever you no longer need data that you've written. This notification lets the controller free space, which can reduce write amplification and increase performance.

The SSD instance store volumes in EC2 high I/O instances provide from tens of thousands to hundreds of thousands of low-latency, random 4 KB random IOPS. Because of the I/O characteristics of SSD devices, write performance can be variable. For more information, see High I/O Instances in the *Amazon EC2 User Guide*.40

The instance store volumes in EC2 high-storage instances provide very high storage density and high sequential read and write performance. For more information, see High Storage Instances in the *Amazon EC2 User Guide*.41

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. For more information on the lifecycle of an EC2 instance, see Instance Lifecycle in the *Amazon EC2 User Guide*.42

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. Note that this usage recommendation also applies to the special-purpose SSD and high-density instance store volumes in the high I/O and high storage instance types.

Scalability and Elasticity

The number and storage capacity of Amazon EC2 local instance store volumes are fixed and defined by the instance type. Although you can’t increase or decrease the number of instance store volumes on a single EC2 instance, this storage is still scalable and elastic; you can scale the total amount of instance store up or down by increasing or decreasing the number of running EC2 instances.

To achieve full storage elasticity, include one of the other suitable storage options, such as Amazon S3, Amazon EFS, or Amazon EBS, in your Amazon EC2 storage strategy.

Security

IAM helps you securely control which users can perform operations such as launch and termination of EC2 instances in your account, and instance store volumes can only be mounted and accessed by the EC2 instances they belong to. Also, when you stop or terminate an instance, the applications and data in its instance store are erased, so no other instance can have access to the instance store in the future.

Access to an EC2 instance is controlled by the guest operating system. If you are concerned about the privacy of sensitive data stored in an instance storage volume, we recommend encrypting your data for extra protection. You can do so by using your own encryption tools, or by using third-party encryption tools available on the AWS Marketplace.[43]

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.

Note that in some cases, a local instance store volume device is attached to an EC2 instance upon launch but must be formatted with an appropriate file system and mounted before use. Also, keep careful track of your block device mappings. There is no simple way for an application running on an EC2 instance to determine which block device is an instance store (ephemeral) volume and which is an EBS (persistent) volume.

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. You can find pricing information for Amazon EC2, Amazon EBS, and data transfer at the Amazon EC2 Pricing web page.[44]

AWS Storage Gateway

AWS Storage Gateway connects an on-premises software appliance with cloud-based storage to provide seamless and secure storage integration between an organization's on-premises IT environment and the AWS storage infrastructure.[45] The service enables you to securely store data in the AWS Cloud for scalable and cost-effective storage. AWS Storage Gateway supports industry-standard storage protocols that work with your existing applications. It provides low-latency performance by maintaining frequently accessed data on-premises while securely storing all of your data encrypted in Amazon S3 or Amazon Glacier. For disaster recovery scenarios, AWS Storage Gateway, together with Amazon EC2, can serve as a cloud-hosted solution that mirrors your entire production environment.

You can download the AWS Storage Gateway software appliance as a virtual machine (VM) image that you install on a host in your data center or as an EC2 instance. Once you've installed your gateway and associated it with your AWS account through the AWS activation process, you can use the AWS Management Console to create gateway-cached volumes, gateway-stored volumes, or a gateway-virtual tape library (VTL), each of which can be mounted as an iSCSI device by your on-premises applications.

With gateway-cached volumes, you can use Amazon S3 to hold your primary data, while retaining some portion of it locally in a cache for frequently accessed data. Gateway-cached volumes minimize the need to scale your on-premises storage infrastructure while still providing your applications with low-latency access to their frequently accessed data. You can create storage volumes up to 32

TiB in size and mount them as iSCSI devices from your on-premises application servers. Each gateway configured for gateway-cached volumes can support up to 20 volumes and total volume storage of 150 TiB. Data written to these volumes is stored in Amazon S3, with only a cache of recently written and recently read data stored locally on your on-premises storage hardware.

Gateway-stored volumes store your primary data locally, while asynchronously backing up that data to AWS. These volumes provide your on-premises applications with low-latency access to their entire datasets, while providing durable, off-site backups. You can create storage volumes up to 1 TiB in size and mount them as iSCSI devices from your on-premises application servers. Each gateway configured for gateway-stored volumes can support up to 12 volumes and total volume storage of 12 TiB. Data written to your gateway-stored volumes is stored on your on-premises storage hardware, and asynchronously backed up to Amazon S3 in the form of Amazon EBS snapshots.

A gateway-VTL allows you to perform offline data archiving by presenting your existing backup application with an iSCSI-based virtual tape library consisting of a virtual media changer and virtual tape drives. You can create virtual tapes in your VTL by using the AWS Management Console, and you can size each virtual tape from 100 GiB to 2.5 TiB. A VTL can hold up to 1,500 virtual tapes, with a maximum aggregate capacity of 150 TiB. Once the virtual tapes are created, your backup application can discover them by using its standard media inventory procedure. Once created, tapes are available for immediate access and are stored in Amazon S3.

Virtual tapes that you need to access frequently should be stored in a VTL. Data that you don't need to retrieve frequently can be archived to your virtual tape shelf (VTS), which is stored in Amazon Glacier, further reducing your storage costs.

Usage Patterns

Organizations are using AWS Storage Gateway to support a number of use cases. These use cases include corporate file sharing, enabling existing on-premises backup applications to store primary backups on Amazon S3, disaster recovery, and mirroring data to cloud-based compute resources and then later archiving it to Amazon Glacier.

Performance

Because the AWS Storage Gateway VM sits between your application, Amazon S3, and underlying on-premises storage, the performance you experience depends upon a number of factors. These factors include the speed and configuration of your underlying local disks, the network bandwidth between your iSCSI initiator and gateway VM, the amount of local storage allocated to the gateway VM, and the bandwidth between the gateway VM and Amazon S3. For gateway-cached volumes, to provide low-latency read access to your on-premises applications, it's important that you provide enough local cache storage to store your recently accessed data. The AWS Storage Gateway documentation provides guidance on how to optimize your environment setup for best performance, including how to properly size your local storage.[46]

AWS Storage Gateway efficiently uses your Internet bandwidth to speed up the upload of your on-premises application data to AWS. AWS Storage Gateway only uploads data that has changed, which minimizes the amount of data sent over the Internet. To further increase throughput and reduce your network costs, you can also use AWS Direct Connect to establish a dedicated network connection between your on-premises gateway and AWS.[47]

Durability and Availability

AWS Storage Gateway durably stores your on-premises application data by uploading it to Amazon S3 or Amazon Glacier. Both of these AWS services store data in multiple facilities and on multiple devices within each facility, being designed to provide an average annual durability of 99.999999999 percent (11 nines). They also perform regular, systematic data integrity checks and are built to be automatically self-healing.

Scalability and Elasticity

In both gateway-cached and gateway-stored volume configurations, AWS Storage Gateway stores data in Amazon S3, which has been designed to offer a very high level of scalability and elasticity automatically. Unlike a typical file system that can encounter issues when storing large number of files in a directory, Amazon S3 supports a virtually unlimited number of files in any bucket. Also, unlike a disk drive that has a limit on the total amount of data that can be stored before you must partition the data across drives or servers, an Amazon S3 bucket can store a virtually unlimited number of bytes. You are able to store any number of objects, and Amazon S3 will manage scaling and distributing redundant copies of your information onto other servers in other locations in the same region, all using Amazon's high-performance infrastructure.

In a gateway-VTL configuration, AWS Storage Gateway stores data in Amazon S3 or Amazon Glacier, providing a virtual tape infrastructure that scales seamlessly with your business needs and eliminates the operational burden of provisioning, scaling, and maintaining a physical tape infrastructure.

Security

IAM helps you provide security in controlling access to AWS Storage Gateway. With IAM, you can create multiple IAM users under your AWS account. The AWS Storage Gateway API enables a list of actions each IAM user can perform on AWS Storage Gateway.[48]

The AWS Storage Gateway encrypts all data in transit to and from AWS by using SSL. All volume and snapshot data stored in AWS using gateway-stored or gateway-cached volumes and all virtual tape data stored in AWS using a gateway-VTL is encrypted at rest using AES-256, a secure symmetric-key encryption standard using 256-bit encryption keys. Storage Gateway supports authentication between your gateway and iSCSI initiators by using Challenge-Handshake Authentication Protocol (CHAP).

Interfaces

The AWS Management Console can be used to download the AWS Storage Gateway VM on-premises or onto an EC2 instance (an AMI that contains the gateway VM image). You can then select between a gateway-cached, gateway-stored, or gateway-VTL configuration and activate your storage gateway by associating your gateway's IP address with your AWS account. All the detailed steps for AWS Storage Gateway deployment can be found in Getting Started in the *AWS Storage Gateway User Guide*.[49]

The integrated AWS CLI also provides a set of high-level, Linux-like commands for common operations of the AWS Storage Gateway service.

You can also use the AWS SDKs to develop applications that interact with AWS Storage Gateway. The AWS SDKs for Java, .NET, JavaScript, Node.js, Ruby, PHP, and Go wrap the underlying AWS Storage Gateway API to simplify your programming tasks.

Cost Model

With AWS Storage Gateway, you pay only for what you use. AWS Storage Gateway has the following pricing components: gateway usage (per gateway per month), snapshot storage usage (per GB per month), volume storage usage (per GB per month), virtual tape shelf storage (per GB per month), virtual tape library storage (per GB per month), retrieval from virtual tape shelf (per GB), and data transfer out (per GB per month). You can find pricing information at the AWS Storage Gateway pricing page.[50]

AWS Snowball

AWS Snowball accelerates moving large amounts of data into and out of AWS using secure Snowball appliances.[51] The Snowball appliance is purpose-built for efficient data storage and transfer. All AWS Regions have 80 TB Snowballs while US Regions have both 50 TB and 80 TB models. The Snowball appliance is rugged enough to withstand an 8.5-G jolt. At less than 50 pounds, the appliance is light enough for one person to carry. It is entirely self-contained, with a power cord, one RJ45 1 GigE and two SFP+ 10 GigE network connections on the back and an E Ink display and control panel on the front. Each Snowball appliance is water-resistant and dustproof and serves as its own rugged shipping container.

AWS transfers your data directly onto and off of Snowball storage devices using Amazon's high-speed internal network and bypasses the Internet. For datasets of significant size, Snowball is often faster than Internet transfer and more cost effective than upgrading your connectivity. AWS Snowball supports importing data into and exporting data from Amazon S3 buckets. From there, the data can be copied or moved to other AWS services such as Amazon EBS and Amazon Glacier as desired.

Usage Patterns

Snowball is ideal for transferring anywhere from terabytes to many petabytes of data in and out of the AWS Cloud securely. This is especially beneficial in cases where you don't want to make expensive upgrades to your network infrastructure or in areas where high-speed Internet connections are not available or cost-prohibitive. In general, if loading your data over the Internet would take a week or more, you should consider using Snowball.

Common use cases include cloud migration, disaster recovery, data center decommission, and content distribution. When you decommission a data center, many steps are involved to make sure valuable data is not lost, and Snowball can help ensure data is securely and cost-effectively transferred to AWS. In a content distribution scenario, you might use Snowball appliances if you regularly receive or need to share large amounts of data with clients, customers, or business associates. Snowball appliances can be sent directly from AWS to client or customer locations.

Snowball might not be the ideal solution if your data can be transferred over the Internet in less than one week.

Performance

The Snowball appliance is purpose-built for efficient data storage and transfer, including a high-speed, 10 Gbps network connection designed to minimize data transfer times, allowing you to transfer up to 80 TB of data from your data source to the appliance in 2.5 days, plus shipping time. In this case, the end-to-end time to transfer the data into AWS is approximately a week, including default shipping and handling time to AWS data centers. Copying 160 TB of data can be completed in the same amount of time by using two 80 TB Snowballs in parallel.

You can use the Snowball client to estimate the time it takes to transfer your data (refer to the *AWS Import/Export User Guide* for more details).[52]

In general, you can improve your transfer speed from your data source to the Snowball appliance by reducing local network use, eliminating unnecessary hops between the Snowball appliance and the workstation, using a powerful computer as your workstation, and combining smaller objects. Parallelization can also help achieve maximum performance of your data transfer. This could involve one or more of the following parallelization types: using multiple instances of the Snowball client on a single workstation with a single Snowball appliance; using multiple instances of the Snowball client on multiple workstations with a single

Snowball appliance; and/or using multiple instances of the Snowball client on multiple workstations with multiple Snowball appliances.

Durability and Availability

Once the data is imported to AWS, the durability and availability characteristics of the target storage applies. Amazon S3 is designed for 99.999999999 percent (11 nines) durability and 99.99 percent availability.

Scalability and Elasticity

Each AWS Snowball appliance is capable of storing 50 TB or 80 TB of data. If you want to transfer more data than that, you can use multiple appliances. For Amazon S3, individual files are loaded as objects and can range up to 5 TB in size, but you can load any number of objects in Amazon S3. The aggregate total amount of data that can be imported is virtually unlimited.

Security

You can integrate Snowball with IAM to control which actions a user can perform.[53] You can give the IAM users on your AWS account access to all Snowball actions or to a subset of them. Similarly, an IAM user that creates a Snowball job must have permissions to access the Amazon S3 buckets that will be used for the import operations.

For Snowball, AWS KMS protects the encryption keys used to protect data on each Snowball appliance. All data loaded onto a Snowball appliance is encrypted using 256-bit encryption.

Snowball is physically secured by using an industry- standard Trusted Platform Module (TPM) that uses a dedicated processor designed to detect any unauthorized modifications to the hardware, firmware, or software.

Snowball is included in the AWS HIPAA compliance program so you can use Snowball to transfer large amounts of Protected Health Information (PHI) data into and out of AWS.[54]

Interfaces

There are two ways to get started with Snowball. You can create an import or export job using the AWS Snowball Management Console or you can use the Snowball Job Management API and integrate AWS Snowball as a part of your data management solution. The primary functions of the API are to create, list, and describe import and export jobs, and it uses a simple standards-based REST web services interface. For more details around using the Snowball Job Management API, see the API Reference documentation.[55]

You also have two ways to locally transfer data between a Snowball appliance and your on-premises data center. The Snowball client, available as a download from the AWS Import/Export Tools page, is a standalone terminal application that you run on your local workstation to do your data transfer.[56] You use simple copy (cp) commands to transfer data, and handling errors and logs are written to your local workstation for troubleshooting and auditing. The second option to locally transfer data between a Snowball appliance and your on-premises data center is the Amazon S3 Adapter for Snowball, which is also available as a download from the AWS Import/Export Tools page. You can programmatically transfer data between your on-premises data center and a Snowball appliance using a subset of the Amazon S3 REST API commands. This allows you to have direct access to a Snowball appliance as if it were an Amazon S3 endpoint. Below is an example of how you would reference a Snowball appliance as an Amazon S3 endpoint when executing an AWS CLI S3 list command. By default, the adapter runs on port 8080, but a different port can be specified by changing the adapter.config file.

```
aws s3 ls --endpoint http://localhost:8080
```

The following example steps you through how to implement a Snowball appliance to import your data into AWS using the AWS Snowball Management Console.

1. To start, sign in to the AWS Snowball Management Console and create a job.
2. AWS then prepares a Snowball appliance for your job.
3. The Snowball appliance is shipped to you through a regional shipping carrier (UPS in all AWS regions except India, which uses Amazon Logistics). You can find your tracking number and a link to the tracking website on the AWS Snowball Management Console.

4. A few days later, the regional shipping carrier delivers the Snowball appliance to the address you provided when you created the job.
5. Next, get ready to transfer your data by downloading your credentials, your job manifest, and the manifest's unlock code from the AWS Management Console, and by downloading the Snowball client. The Snowball client is the tool that you'll use to manage the flow of data from your on-premises data source to the Snowball appliance.
6. Install the Snowball client on the computer workstation that has your data source mounted on it.
7. Move the Snowball appliance into your data center, open it, and connect it to power and your local network.
8. Power on the Snowball appliance and start the Snowball client. You provide the IP address of the Snowball appliance, the path to your manifest, and the unlock code. The Snowball client decrypts the manifest and uses it to authenticate your access to the Snowball appliance.
9. Use the Snowball client to transfer the data that you want to import into Amazon S3 from your data source into the Snowball appliance.
10. After your data transfer is complete, power off the Snowball appliance and unplug its cables. The E Ink shipping label automatically updates to show the correct AWS facility to ship to. You can track job status by using Amazon SNS, text messages, or directly in the console.
11. The regional shipping carrier returns the Snowball appliance to AWS.
12. AWS gets the Snowball appliance and imports your data into Amazon S3. On average, it takes about a day for AWS to begin importing your data into Amazon S3, and the import can take a few days. If there are any complications or issues, we contact you through email.

Once the data transfer job has been processed and verified, AWS performs a software erasure of the Snowball appliance that follows the National Institute of Standards and Technology (NIST) 800-88 guidelines for media sanitization.

Cost Model

With Snowball, as with most other AWS services, you pay only for what you use. Snowball has three pricing components: service fee (per job), extra day charges as required (the first 10 days of onsite usage are free), and data transfer. For the destination storage, the standard Amazon S3 storage pricing applies. You can find pricing information at the AWS Snowball Pricing page.[57]

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.[58] 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. To get a better understanding of your end user usage patterns, you can use Amazon CloudFront reports.[59]

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.[60] [61] 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.

Scalability and Elasticity

Amazon CloudFront is designed to provide seamless scalability and elasticity. You can easily start very small and grow to massive numbers of global connections. With Amazon CloudFront, you don't need to worry about maintaining expensive web server capacity to meet the demand from potential traffic spikes for your content. The service automatically responds as demand spikes and fluctuates for your content, without any intervention from you.

Amazon CloudFront also uses multiple layers of caching at each edge location and collapses simultaneous requests for the same object before contacting your origin server. These optimizations further reduce the need to scale your origin infrastructure as your website becomes more popular.

Security

Amazon CloudFront is a very secure service to distribute your data. It integrates with IAM so that you can create users for your AWS account and specify which Amazon CloudFront actions a user (or a group of users) can perform in your AWS account.

You can configure Amazon CloudFront to create log files that contain detailed information about every user request that Amazon CloudFront receives. These access logs are available for both web and RTMP distributions.[62] Additionally, Amazon CloudFront integrates with Amazon CloudWatch metrics so that you can monitor your website or application.[63]

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. For example, you can enable or disable distributions, configure CNAMEs, and enable end-user logging using the console. 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.

Although there are no long-term contracts or required minimum monthly commitments, CloudFront offers an optional reserved capacity plan that gives you the option to commit to a minimum monthly usage level for 12 months or longer and in turn receive a significant discount. You can find pricing information at the Amazon CloudFront pricing page.[64]

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.

Contributors

The following individuals contributed to this document:

- Darryl S. Osborne, Solutions Architect, Amazon Web Services
- Shruti Worlikar, Solutions Architect, Amazon Web Services
- Fabio Silva, Solutions Architect, Amazon Web Services

References and Further Reading

AWS Storage Services

- Amazon S3[65]
- Amazon Glacier[66]
- Amazon EFS[67]
- Amazon EBS[68]
- Amazon EC2 Instance Store[69]
- AWS Storage Gateway[70]
- AWS Snowball[71]
- Amazon CloudFront[72]

Other Resources

- AWS SDKs, IDE Toolkits, and Command Line Tools[73]
- Amazon Web Services Simple Monthly Calculator[74]
- Amazon Web Services Blog[75]
- Amazon Web Services Forums[76]
- AWS Free Usage Tier[77]
- AWS Case Studies[78]

Notes

[1] https://aws.amazon.com/s3/

[2] https://docs.aws.amazon.com/AmazonS3/latest/dev/crr.html

[3] http://docs.aws.amazon.com/AmazonS3/latest/dev/uploadobjusingmpu.html

[4] http://docs.aws.amazon.com/AmazonS3/latest/dev/access-control-overview.html#access-control-resources-manage-permissions-basics

[5] http://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

[6] http://docs.aws.amazon.com/AmazonS3/latest/dev/UsingClientSideEncryption.html

[7] http://docs.aws.amazon.com/AmazonS3/latest/dev/Versioning.html#MultiFactorAuthenticationDelete

[8] http://docs.aws.amazon.com/AmazonS3/latest/dev/ServerLogs.html

[9] http://aws.amazon.com/sns/

[10] http://aws.amazon.com/sqs/

[11] http://aws.amazon.com/lambda/

[12] http://aws.amazon.com/free/

[13] http://aws.amazon.com/s3/pricing/

[14] http://aws.amazon.com/glacier/

[15] http://docs.aws.amazon.com/amazonglacier/latest/dev/uploading-archive-mpu.html

[16] http://docs.aws.amazon.com/amazonglacier/latest/dev/downloading-an-archive.html#downloading-an-archive-range

[17] https://aws.amazon.com/iam/

[18] http://aws.amazon.com/cloudtrail/

[19] http://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

[20] http://aws.amazon.com/glacier/pricing/

[21] http://aws.amazon.com/efs/

22 http://docs.aws.amazon.com/efs/latest/ug/how-it-works.html

23 http://docs.aws.amazon.com/efs/latest/ug/monitoring-cloudwatch.html#efs-metrics

24 http://docs.aws.amazon.com/efs/latest/ug/mounting-fs.html

25 http://docs.aws.amazon.com/efs/latest/ug/mounting-fs-mount-cmd-general.html

26 http://docs.aws.amazon.com/efs/latest/ug/security-considerations.html

27 http://aws.amazon.com/efs/pricing/

28 http://aws.amazon.com/ebs/

29 https://aws.amazon.com/ebs/previous-generation/

30 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#monitoring_burstbucket

31 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#monitoring_burstbucket

32 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSVolumeTypes.html#monitoring_burstbucket

33 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ebs-initialize.html

34 https://aws.amazon.com/ebs/details/

35 https://aws.amazon.com/kms/

36 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/EBSEncryption.html

37 http://aws.amazon.com/ebs/pricing/

38 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html

39 http://docs.aws.amazon.com/AWSEC2/latest/WindowsGuide/block-device-mapping-concepts.html

40 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/i2-instances.html

[41] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/high_storage_instances.html

[42] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-lifecycle.html

[43] https://aws.amazon.com/marketplace

[44] http://aws.amazon.com/ec2/pricing/

[45] http://aws.amazon.com/storagegateway/

[46] http://docs.aws.amazon.com/storagegateway/latest/userguide/WhatIsStorageGateway.html

[47] http://aws.amazon.com/directconnect/

[48] http://docs.aws.amazon.com/storagegateway/latest/userguide/AWSStorageGatewayAPI.html

[49] http://docs.aws.amazon.com/storagegateway/latest/userguide/GettingStarted-common.html

[50] http://aws.amazon.com/storagegateway/pricing/

[51] https://aws.amazon.com/importexport/

[52] http://aws.amazon.com/importexport/tools/

[53] http://docs.aws.amazon.com/AWSImportExport/latest/DG/auth-access-control.html

[54] https://aws.amazon.com/about-aws/whats-new/2016/11/aws-snowball-now-a-hipaa-eligible-service/

[55] https://docs.aws.amazon.com/AWSImportExport/latest/ug/api-reference.html

[56] https://aws.amazon.com/importexport/tools/

[57] http://aws.amazon.com/importexport/pricing/

[58] http://aws.amazon.com/cloudfront/pricing/

[59] http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/reports.html

60 http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Invalidation.html

61 http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/ReplacingObjects.html

62 http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html

63 http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/monitoring-using-cloudwatch.html

64 http://aws.amazon.com/cloudfront/pricing/

65 http://aws.amazon.com/s3/

66 http://aws.amazon.com/glacier/

67 http://aws.amazon.com/efs/

68 http://aws.amazon.com/ebs/

69 http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/InstanceStorage.html

70 http://aws.amazon.com/storagegateway/

71 http://aws.amazon.com/snowball

72 http://aws.amazon.com/cloudfront/

73 http://aws.amazon.com/tools/

74 http://calculator.s3.amazonaws.com/index.html

75 https://aws.amazon.com/blogs/aws/

76 https://forums.aws.amazon.com/index.jspa

77 http://aws.amazon.com/free/

78 http://aws.amazon.com/solutions/case-studies/