**5**

**Storage in AWS – Choosing the Right Tool for the Job**

Storage is a critical and foundational service for any cloud provider. If this service is not implemented in a durable, available, efficient, low-latency manner, it doesn’t matter how many other excellent services are offered.

File, block, and object storage are at the core of many applications. In *Chapter 7*, *Selecting the Right Database Service*, we will learn about other storage services focused on databases. However in this chapter, we will focus on basic file and object storage.

In this chapter, we will first look at Amazon EBS, EFS, and S3. We will then look at the difference between block storage and object storage. We will also look at versioning in Amazon S3 and explore Amazon S3 best practices.

In this chapter, we will cover the following topics:

* Understanding local storage with Amazon **Elastic Block Store** (**EBS**)
* Investigating file storage with Amazon **Elastic File System** (**EFS**)
* Using Amazon FSx to manage file systems
* Versioning in Amazon S3
* Choosing the right cloud storage type
* Exploring Amazon S3 best practices
* Building hybrid storage with AWS Storage Gateway and AWS Backup

Moving storage workloads to the cloud has been one of the main ways to address strategic priorities, such as increasing an organization’s agility, accelerating its ability to innovate, strengthening security, and reducing cost. Let’s learn how all of this can be achieved in AWS.

**Understanding Amazon Elastic Block Store**

Block storage is a foundational storage technology that has been around since the early days of computing. The hard drive in your laptop, the memory in your mobile phone, and all other forms of data storage, from USB thumb drives to storage arrays that organizations place in their data centers, are all based on block storage.

Persistent block storage that can be used with Amazon EC2 instances is provided by **Amazon Elastic Block Store**(**EBS**). When using EC2, you have the option to use local instance storage or EBS for block storage:

* **Instance storage** is great for high-performance (over 80K IOPS and over 1,750 MB/s throughput) and low-latency (under 1 ms) applications. However, instance storage is ephemeral, which means when you stop, hibernate, or terminate an EC2 instance, every block of storage in the instance store is reset. Therefore, do not rely on instance storage for valuable, long-term data.
* **EBS volumes** provide excellent performance and persistent storage. EBS allows your customer to correctly size their instance for the memory and CPU they need, relying on EBS for their storage, which they can independently size on capacity, IOPS, or throughput.
* **SAN** (**Storage Area Network**) in the cloud – With io2 Block Express, you can now achieve SAN-like performance in the cloud.

In simple terms, Amazon EBS is a hard drive for a server in AWS. One advantage of Amazon EBS over many typical hard drives is that you can easily detach it from one server and attach it to another server using software commands. Usually, with other servers outside of AWS, this would require physically detaching the hard drive and physically attaching it to another server.

When using Amazon EBS, data is persisted. This means that data lives even after the server is shut down. Like other services, Amazon EBS provides high availability and durability.

Amazon EBS should not be confused with the instance store available in EC2 instances. EC2 instance stores deliver ephemeral storage for EC2 instances. One of the use cases for EC2 instance stores would be any data that does not need to be persisted, such as the following:

* Caches
* Buffers
* Temporary files
* Temporary variables

If data needs to be stored permanently, the following Amazon EBS options are available.

**General-purpose Solid-State Devices (SSDs)**

General-purpose **Solid-State Device** (**SSD**) storage provides a solid balance of cost and performance. It can be applied to a wide array of use cases, such as the following:

* Virtual desktops
* Development and staging environments
* Application development

There are two types of general-purpose SSD volume available, gp2 and gp3. They come in a volume size of 1 TB to 16 TB with **input/output operations per second** (**IOPS**) up to 16,000. IOPS is a useful way to measure disk performance. The gp3 is the latest generation of SSD volume, which costs 20% less than the gp2 volume and provides four times more throughput up to 1,000 MiB/s. A gp3 volume provides 128MiB/s throughput and 3,000 IOPS performance which can scale up to 16,000 IOPS and 1,000 MiB/s if needed with nominal charges.

**Provisioned IOPS SSD**

Amazon EBS storage volumes with Provisioned IOPS SSD are intended for low-latency workloads that require high IOPS and throughput. They are designed to provide the highest performance and are ideal for critical and IOPS-intensive applications. This type of storage is ideally suited for mission-critical applications that require a high IOPS performance, such as the following:

* Business applications
* Production databases, including SAP HANA, MS SQL Server, and IBM DB2

There are two types of provisioned IOPS SSDs available, io1 and io2. They come in a volume size of 4 TB to 16 TB with IOPS up to 64,000. The io2 is a next-generation volume and offers a consistent baseline performance of up to 500 IOPS/GB compared to io1, which offers 50 IOPS/GB performance. Improving further on performance, AWS launched the “io2 Block Express” volume, which comes in volume sizes of 4 TB to 64 TB with IOPS up to 256,000.

**Throughput Optimized HDD**

Throughput Optimized **HDDs** (**Hard Disk Drives**) offer excellent value, providing a reasonable cost for workloads that call for high performance and have high throughput requirements. Typical use cases include:

* Big data applications
* Log processing
* Streaming applications
* Data warehouse applications

Throughput Optimized HDDs, also known as st1, come in volume sizes of 125 GB to 16TB with 500 IOPS per volume. An st1 volume baseline performance scales with the size of the volume. Here, instead of measuring in IOPS, we’re talking throughput in MB/second. The baseline is 40MB/second per TB provisioned, up to 500MB/second. These are not designed for boot volumes and you have a minimum capacity of 125 GiB. But you can go up to 16 TiB.

**Cold HDD**

This type of storage is normally used for applications that require optimizing costs with large volumes of data—typically, data that needs to be accessed infrequently. Cold HDD also goes by sc1 and comes in volume sizes 125 GB to 16TB with 250 IOPS per volume.

**IMPORTANT NOTE**

This would be a good time to note the following. EC2 instances are virtualized, and there isn’t a one-to-one relationship between servers and EC2 instances. Similarly, when you use EBS storage, a single physical storage device is not assigned to you by AWS; instead, you get a slice of several devices that store the data in a distributed fashion across data centers to increase reliability and availability. You can attach an EBS volume to an EC2 instance in the same AZ only; however, you can use a snapshot to create multiple volumes and move across AZs.

In addition, Amazon **Elastic Block Store** (**EBS**) volumes are replicated transparently by design. Therefore, it is unnecessary to provide extra redundancy by setting up RAID or other redundancy strategies.

Amazon EBS volumes are by default highly available, durable, and reliable. This redundancy strategy and multiple server replication are built into the base price of Amazon EBS volumes. Amazon EBS volume files are mirrored across more than one server within an **Availability Zone** (**AZ**). This will minimize data loss. More than one device will have to fail simultaneously for data loss to occur. Amazon EBS volumes will also self-heal and bring in additional healthy resources if a disk fails.

Amazon EBS volumes offer at least twenty times more reliability than traditional commodity devices. Let’s learn about EBS Snapshots.

**Amazon EBS Snapshots**

Amazon EBS also provides a feature to quickly and automatically create snapshots. The backups of data volumes can be performed incrementally. For example, if you have a device with 100 GB of data, the first day the snapshot is created, the snapshot will have to reflect all 100 GB. If, the next day, 5 GB of additional data is added, when the next snapshot is taken, EBS is smart enough to realize that it only needs to account for the new data. It can use the previous snapshot with the latest backup to recreate the full picture. This incremental snapshot strategy will translate into lower storage costs.

Snapshots can be compressed, mirrored, transferred, replicated, and managed across multiple AWS AZs using the Amazon Data Lifecycle Manager.

Amazon EBS Snapshots are stored as Amazon S3 objects. Amazon EBS Snapshots are accessed using the Amazon EBS API, and they cannot be accessed directly by users. Snapshots are stored as **Amazon Machine Images** (**AMIs**), and therefore, they can be leveraged to launch an EC2 instance.

While optimizing the cost, the first place to look is into unused EBS Snapshots. People often take EBS Snapshots even for the dev/test environment and never access them, causing additional costs. You should always use lifecycle management to delete unused EBS Snapshots as needed. For example, you can set up a policy to delete any EBS Snapshots which are tagged as dev/test and are older than six months.

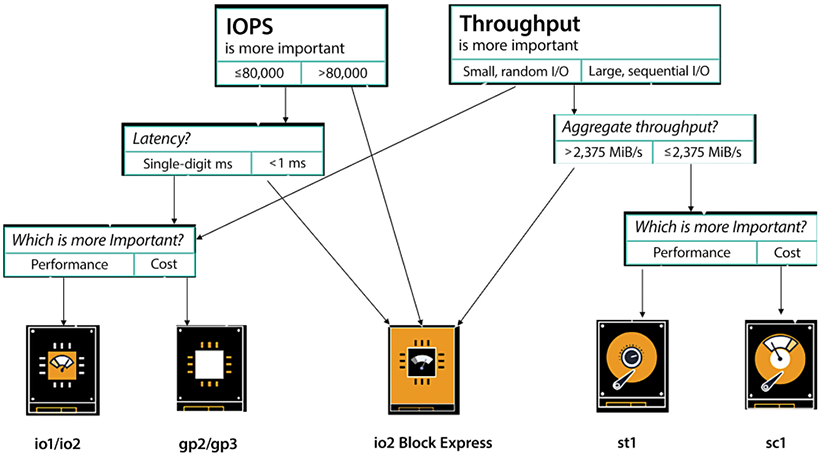
We have learned about many EBS volumes; now let’s understand how to choose the right EBS volume.

**Choosing the right EBS volume**

As you learned in the previous section, EBS offers 4 volume types in two different buckets: SSD and HDD. The first step in helping you size your storage is to understand whether your workload is sequential or random I/O:

* **SSD** (gp2, gp3, io1, io2, and io2 Block Express) is great for random I/O applications such as boot volumes and databases (MySQL, SQL, PostgreSQL, Oracle, Cassandra, MongoDB, SAP, etc.). Performance is measured on disk I/O (IOPS).
* **HDD** (st1 and sc1) is great for sequential I/O applications like EMR/Hadoop, Kafka, Splunk, media streaming, and logs, as well as any colder data that is infrequently accessed. Performance is measured on throughput (MB/s).

From there, you can use AWS documentation (<https://aws.amazon.com/ebs/volume-types/>) to help decide which specific volume will best suit your needs and give the best price for performance. Here is a decision tree to help in the process:



*Figure 5.1: Decision tree to select the EBS volume*

To understand the approach for choosing the right EBS volume, let’s dive a little more into the two SSD-backed products, gp3 and io2. Starting with the **gp3 volume**, it is a volume type you can spend a lot of time with analyzing data, analyzing workloads, and developing it to achieve the performance sweet spot for 70 to 80% of the workloads. gp3 satisfies nearly all workloads and is designed to be a truly general-purpose volume. That’s why it has the name general purpose, so if you don’t know which volume type to use, it is highly recommended that you start with gp3. Additionally, with gp3, you have the ability to provision more IOPS and more throughput when you need it without being dependent on storage size.

The low latency of gp3, which is measured in single-digit milliseconds, makes it well suited for applications that are sensitive to latency, such as interactive applications, boot volumes, development and testing environments, burst databases, and various other use cases. But what about sub-millisecond latency? Now, if you are looking for very high performance and low latency, use **io2 Block Express volumes,** which offer 4 times the performance of io1, up to 256,000 IOPS, 4,000 MB/s throughput, and you can provision up to 4x more storage – up to 64 TB.

Now, we move on to the media workload, which is for your rendering farms, transcoding, encoding, and any sort of streaming product. Here, you typically have a higher throughput requirement, mostly sequential and fairly sustained, especially when you have a pretty substantial render job. For these kinds of workloads, **Throughput Optimized HDDs** (also known as **st1s**) might be a good fit. These are good for large blocks, high throughput, and sequential workloads.

SC1 is a low-cost storage option that is ideal for storing infrequently accessed data, such as data that is accessed once a month or less. It is a good choice for cold storage or for storing data that is not accessed very often. ST1, on the other hand, is a low-cost storage option that is designed for storing frequently accessed data, such as data that is accessed on a daily or weekly basis. It is a good choice for data that is accessed frequently, but not in real time.

In summary, the main difference between SC1 and ST1 is their intended use cases. SC1 is designed for infrequently accessed data, while ST1 is designed for frequently accessed data.

In a nutshell, with EBS, you get persistent storage, which enables you to stop and start your instances without losing your storage and you can build cost-effective point-in-time snapshots of the volumes that you are storing in S3. For security, EBS provides built-in encryption so you don’t need to manage your own. For monitoring, EBS has better volume monitoring with EBS CloudWatch metrics. It is designed for 99.999% availability and is highly durable with an annual failure rate of between 0.1% and 0.2%.

This wraps up what we wanted to say about the Amazon EBS service. Now, let’s move on to another important service—Amazon Elastic File System.

**Investigating Amazon Elastic File System (EFS)**

Amazon EFS implements an elastic, fully-managed **Network File System** (**NFS**) that can be leveraged by other AWS Cloud services and on-premises infrastructure. Amazon EFS natively integrates with the complete family of AWS compute models, and can scale as needed to provide parallel, shared access to thousands of Amazons EC2 instances as well as AWS container and serverless compute models from AWS Lambda, AWS Fargate, Amazon **Elastic Container Service** (**ECS**), and Amazon **Elastic Kubernetes Service** (**EKS**).

The main difference between EBS and EFS is that several EC2 instances can be mounted to an EFS volume simultaneously, while an EBS volume can be attached to only one EC2 instance. Amazon EFS provides shared file storage that can elastically adjust on demand to expand or shrink depending on how much space your workloads require. It can grow and shrink as you add and remove files. Other than that, the structure will be like it is with Amazon EBS. Amazon EFS provides a typical file storage system where files can be organized into directories and subdirectories.

Common use cases for EFS volumes include:

* Web serving and content management systems for WordPress, Drupal, Moodle, Confluence, and OpenText.
* Data science and analytics for TensorFlow, Qubole, and Alteryx.
* Media processing, including video editing, sound design, broadcast processing, studio production, and rendering. This often depends on shared storage to manipulate large files.
* Database backups for Oracle, Postgres, Cassandra, MongoDB, CouchDB, and SAP HANA.
* Hosting CRM applications that require hosting within the AWS data center but need to be managed by the AWS customer.

The following are the key benefits of using EFS:

* EFS is **elastic**, automatically scaling up or down as you add or remove files, and you pay only for what you use. Your performance automatically scales with your capacity. By the way, EFS file systems scale to petabytes in size.
* EFS is also highly available and designed to be highly durable. AWS offers a 4 9s availability SLA (99.99% availability) and is designed for 11 9s of data durability(which means it delivers 99.999999999% durability across multiple AZs).

To achieve these levels of availability and durability, all files and directories are redundantly stored within and across multiple AZs. EFS file systems can withstand the full loss of a single AZ, while still providing the same quality of service in the other AZs.

* EFS is serverless; you don’t need to provision or manage any infrastructure or capacity. And as your workload scales up, so does your file system, automatically accommodating any additional storage or connection capacity that you need.
* EFS file systems support up to tens of thousands of concurrent clients, no matter the type. These could be traditional EC2 instances, containers running in one of your self-managed clusters or in one of the AWS container services, Elastic Container Service, Elastic Kubernetes Service, and Fargate – or in a serverless function running in AWS Lambda. You can also access your EFS file systems from on-premises through AWS Direct Connect and AWS VPN.
* In terms of performance, EFS file systems provide low, consistent latencies (in the single digit millisecond range for active file system workloads), and can scale to tens of GB/s of throughput and support over 500,000 IOPS.
* Finally, EFS storage classes provide you with automatic cost-optimization and help you achieve an optimal price/performance blend for your workloads. Files that you aren’t using frequently will be automatically moved from the Standard storage class to the lower-cost EFS **Infrequent Access** (**IA**) storage class, completely transparently to users and applications. IA costs 92% less than Standard.

You can learn more about EFS by visiting the AWS website: <https://aws.amazon.com/efs/>.

While EFS provides generic file system storage, there is a need for file system-specific storage to handle the workload optimized for a particular file system. For that, AWS provides Amazon FSx.

**Using Amazon FSx to manage file systems**

Amazon FSx is a managed file storage service offered by AWS. It is a fully managed service that allows users to store and access files over the internet using the industry-standard **network file system** (**NFS**) and **server message block** (**SMB**) protocols.

FSx is designed to provide high performance and high availability for file storage, making it a good choice for applications that require fast access to files or data. It supports a wide range of workloads, including big data analytics, content management, and video editing.

FSx offers two main types of file storage: FSx for Lustre and FSx for Windows File Server. FSx for Lustre is a high-performance file system that is optimized for workloads that require low-latency access to data, such as high-performance computing and machine learning.

FSx for Windows File Server is a fully managed native Microsoft Windows file system that is compatible with the SMB protocol and is ideal for applications that require seamless integration with Microsoft Windows.

Overall, Amazon FSx is a fully managed file storage service that provides high performance and high availability for file storage, making it a good choice for applications that require fast access to files or data.

You may want to use Amazon FSx for the following use cases:

* Migrating file system-specific workloads to the cloud.
* Running ML and analytics applications using **high-performance computing** (**HPC**) with FSx Lustre.
* Hosting and editing media and entertainment workloads where high performance is needed.
* Creating a backup of file system-specific storage in the cloud for business continuity strategy.

In this section, you learned about Amazon EFS/FSx file storage, which can be compared to **NAS** (**Network Attached Storage**) in the on-premises data center. Further, you went into details about Amazon EBS, which is block storage and can be compared with **SAN** (**Storage Area Network**) in an on-premise environment. Let’s now learn about the object storage system in AWS called Amazon **Simple Storage Service** (**S3**).

**Learning about Amazon Simple Storage Service (S3)**

Amazon S3 was the first AWS service launched 16 years ago on Pi Day, March 14, 2006. After the launch of S3, AWS also launched many other services to complement S3. S3 is durable, highly available, and very scalable online storage. S3 comes in various tiers including:

* S3 Standard
* Amazon S3 Intelligent-Tiering
* Amazon S3 Standard-IA
* Amazon S3 One Zone-IA
* Amazon S3 Glacier
* Amazon S3 Glacier Deep Archive

While we will explore each of the above tiers in detail in the subsections below, let’s briefly explore some of the more common attributes that apply to multiple S3 service tiers:

* **Durability**: Data is stored durably, with 11 9s across a minimum of 3 AZs to provide resiliency against an AZ failure for S3 Standard, S3 Intelligent-Tiering, and S3 Standard-IA. S3 One Zone-IA has11 9s across 1 Availability Zone. This means that if you stored 10K objects, on average, you would lose 1 object every 10M years with S3.
* **Availability**: S3 Standard is designed for 99.99% availability, with monetary penalties introduced at 99.9% per the S3 SLA. S3 Intelligent-Tiering and S3 Standard-IA are built to be 99.9% available, while S3 One Zone-IA is built to be 99.5% available. These three classes are backed by a 99% availability SLA. There are no scheduled maintenance windows – the service is maintained, upgraded, and scaled as it operates. 99.99% availability means that the service is down for less than 52 minutes a year.
* **Utility pricing**: There is no upfront capital expenditure, and pay for what you use pricing (100% utilization). Volume discounts as usage scales.
* **Scalability**: Instant scalability for storage and delivery.

Let’s analyze the various S3 tiers in detail.

**S3 Standard**

When Amazon launched the S3 service, it was simply called Amazon S3. Amazon now offers various object storage services, and they all use the S3 moniker, so Amazon has renamed Amazon S3 as Amazon S3 Standard.

S3 Standard delivers highly performant, available, and durable storage for data that will be accessed frequently. S3 Standard has low latency, high performance, and high scalability. S3 Standard is suited for a long list of use cases, including the following:

* Websites with dynamic content – You can host website HTML pages in S3 and directly attach them to your domain which will be very low cost as you don’t need any server to host your website. Further, you can supplement your website with dynamic content like videos and images by putting them into S3.
* Distribution of content – To make your website fast for a global audience, you can host your content in S3, which will be cached in an edge location by AWS’s content distribution network service called AWS CloudFront.
* Data analytics and processing – You can host a large amount of data and scale it on demand for your analytics and machine learning needs.
* Mobile and gaming applications – You can supplement your application by storing all application heavy data like images and videos in S3 and load them on-demand to improve application performance.

S3 Standard can be used to persist many types of objects, such as Plaintext, HTML, JSON, XML, AVRO, Parquet, and ORC files. S3 Standard is one of the most popular services, addressing a wide range of use cases. You can learn more about S3 by visiting the AWS page here - <https://aws.amazon.com/s3/>.

Let’s now look at another service in the S3 family, Amazon S3 Intelligent-Tiering, and let’s learn what makes it intelligent.

**Amazon S3 Intelligent-Tiering**

The S3 Intelligent-Tiering storage service can reduce expenses by systematically moving files to use the most cost-effective way to store data while having no impact on operations or performance. It can do this by keeping the files in two tiers:

* An optimized tier for frequent access
* An optimized tier for infrequent access that has a lower cost

Amazon S3 constantly scans access patterns of files and transfers files that have not been accessed. If a file has not been accessed for 30 days straight, it is moved to the infrequent access tier. If a file in the infrequent access tier is retrieved, it is again transferred to the frequent access tier.

With the S3 Intelligent-Tiering storage class, the additional cost comes from the monitoring charge. There are no fees for retrieval, and there are no additional file transfer fees when objects are transferred between the tiers. S3 Intelligent-Tiering is a good solution when we know that data will be needed for a long time but we are uncertain about how often this data will be accessed.

S3 services can be enabled granularly up to the object level. For example, a given bucket can have one object that uses S3 Standard, another using S3 Intelligent-Tiering, one more with S3 Standard-IA, and one with S3 One Zone-IA (we are going to cover these two other services in the next few sections).

Note: To take advantage of auto-tiering capability, it is recommended to aggregate objects smaller than 128KB to meet the minimum size requirement. This enables the objects to be automatically tiered and reduces storage costs. Smaller objects, which are not monitored, are always charged at the rates of the Frequent Access tier, without any additional monitoring or automation charges.

So, what if we know that the data we create and store will be infrequently accessed? Amazon offers a service ideally suited for that, which will also be cheaper than Amazon S3 Standard.

**Amazon S3 Standard-IA (Infrequent Access)**

Depending on the use case for your data, this storage class might be the ideal solution for your data storage needs. The data can be accessed at the same speed as S3 Standard, with some tradeoffs on the resiliency of the data. For example, if you are storing monthly payroll data, it is going to be accessed most in the last week of each month, so it’s better to use S3 Standard-**IA** (**Infrequent Access**) to save cost.

S3 Standard-IA offers a similar profile to the Standard service but with a lower cost for storage and with a retrieval fee billed per GB. Combining low cost with high performance makes S3 Standard-IA a well-suited option for use cases such as backups, snapshots, long-term storage, and a file repository for disaster recovery. S3 Lifecycle policies could be used to move files between storage classes without any coding needed automatically.

It is recommended that you configure your Amazon S3 to manage your items and keep them cost-effectively stored throughout their lifecycle. An S3 Lifecycle configuration is a set of rules that specify how Amazon S3 handles a collection of objects. By using S3 Lifecycle configuration rules, you can instruct Amazon S3 to transition objects to more cost-effective storage classes, or to archive or delete them automatically after a specified period of time.

You can learn more details on setting up the S3 Lifecycle by visiting the AWS user document here - <https://docs.aws.amazon.com/AmazonS3/latest/userguide/how-to-set-lifecycle-configuration-intro.html>.

So, what if your data is not that critically important and you are willing to give up some durability in exchange for a cheaper alternative? Amazon has a service that fits that criteria. We’ll learn about it in the next section.

**Amazon S3 One Zone-IA**

A better name for S3 Standard-IA might be *S3 Standard-IA; that is not critical*. Like the previous service, S3 Standard-IA, S3 One Zone-IA can be used for files that need to be retrieved with less frequency but need rapid access. This service is cheaper than S3 Standard because instead of storing data in three AZs, S3 One Zone-IA persists data in only one AZ with the same durability.

S3 One Zone-IA is a good solution for use cases that don’t need the reliability of S3 Standard or S3 Standard-IA and therefore get a lower price. The reliability is still high, and it still has duplication, but this duplication is not done across AZs. It’s a suitable option for storing files such as backup files and data that can be quickly and easily recreated. It can also be a cost-effective way to store data that has been copied from another AWS Region with S3 Cross-Region Replication.

**IMPORTANT NOTE**

Keep in mind that these files will be unavailable and potentially destroyed if an AZ goes down or is destroyed. So, it should only be used with files that are not mission-critical.

S3 One Zone-IA delivers similar high throughput, durability, and speed to S3 Standard, coupled with an inexpensive retrieval cost.

So far, we have looked at services that allow us to access the data immediately. What if we are willing to give up that immediate accessibility in exchange for an even cheaper service? Amazon S3 Glacier fits that bill. We’ll learn about it in the next section.

**Amazon S3 Glacier**

When it’s known that a file will not be needed immediately, S3 Glacier is a good option. S3 Glacier is a secure, durable class for data archiving. It is significantly cheaper than S3 Standard, but it will take longer to retrieve an S3 Glacier file. Data can be stored on S3 Glacier at a cost that would be competitive with an on-premises solution. Within S3 Glacier, the options and pricing are flexible.

Amazon S3 Glacier storage classes are designed specifically for archiving data, offering superior performance, retrieval flexibility, and cost-effective storage options in the cloud. These storage classes provide virtually unlimited scalability, as well as 11 nines of data durability, ensuring the safety and reliability of your data over the long term.

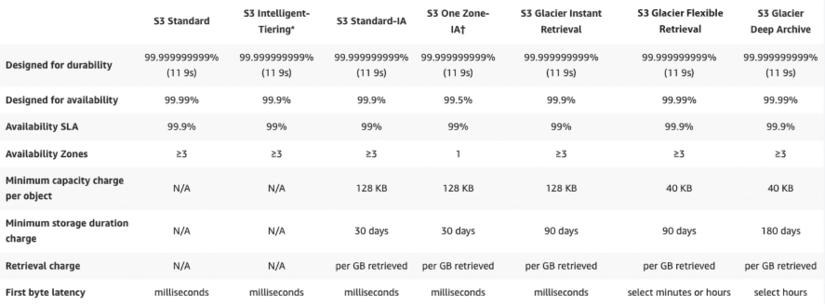
S3 Glacier provides three different storage classes: S3 Glacier Instant Retrieval, S3 Glacier Flexible Retrieval, and S3 Glacier Deep Archive. **S3 Glacier Flexible Retrieval** is the base storage class of S3 Glacier and is designed for long-term storage of data that is accessed infrequently. It is a low-cost storage option that is ideal for storing data that is accessed once a year or less. You can use this option if you need to retrieve data in minutes to 12 hours time. **S3 Glacier Deep Archive** is the lowest-cost storage class of S3 Glacier and is designed for long-term storage of data that is accessed once a year or less.

It is the lowest-cost storage option provided by AWS and is ideal for storing data that is rarely accessed and does not need to be retrieved quickly. You can use this option if you can wait for data for 12 hours to 24 hours time to retrieve data. **S3 Glacier Instant Retrieval** is a storage class that allows you to query and analyze data stored in S3 Glacier without having to retrieve the entire data set. It allows you to run SQL queries on your data in S3 Glacier and only retrieves the data that is needed for the query, providing faster and more cost-effective access to your data. You should use this option if you need to retrieve data immediately in milliseconds, however, the faster data retrieval time comes with higher cost. These storage classes are designed for different use cases and provide low-cost options for storing data that is infrequently accessed. You can choose them as per your workload requirements.

S3 Glacier Deep Archive is Amazon S3’s cheapest option for object storage. It enables long-term storage. It is suited for files that are only going to be retrieved occasionally. It is designed for customers that are required to keep data for seven years or longer to meet regulatory compliance regulations such as in the financial industry, healthcare industry, and government agencies.

In many cases, heavy penalties can accrue if these rules are not adequately followed. Other good use cases are backup and disaster recovery. Many customers are using this service instead of magnetic tape systems. S3 Glacier Deep Archive can be used in conjunction with Amazon S3 Glacier, allowing data to be retrieved faster than the Deep Archive service.

In the following figure, we have a summary of the profile of storage classes and how they compare to each other:



*Figure 5.2: Summary of storage class features*

In the above summary, the cost of storage gets cheaper as we move from left to right. This means that S3 Standard has the highest cost of storage while S3 Glacier Deep Archive is the cheapest.

In this section, you have learned that AWS has a vast number of offerings for your storage use cases. Depending on how quickly the data needs to be retrieved and how durably it needs to be stored, the costs will vary and allow for savings if high durability and fast retrieval are not required. Sometimes you want to change data in runtime. For that, AWS launched a new feature called S3 Object Lambda. Let’s learn more about it.

**Managing data with S3 Object Lambda**

Amazon S3 Object Lambda is a feature of Amazon S3 that allows users to run custom code on objects stored in S3. With S3 Object Lambda, you can define custom actions that are triggered when an object is created, updated, or deleted in S3. These actions can be used to perform a variety of tasks, such as automatically resizing images, creating thumbnails, or transcoding videos.

To use S3 Object Lambda, you first need to write custom code that defines the actions you want to perform. This code can be written in a variety of languages, such as Node.js, Python, or Java, and can be run on AWS Lambda, a serverless compute service.

Once you have written your custom code, you can create a Lambda function and attach it to an S3 bucket. When objects are created, updated, or deleted in the S3 bucket, the Lambda function will be triggered and will perform the actions defined in your code.

S3 Object Lambda is a powerful feature that can be used to automate a variety of tasks and improve the management of data stored in S3. It allows you to perform custom actions on objects in S3 and can save time and effort when working with large amounts of data.

Sometimes you want to create multiple copies of an object, in which case S3 allows versioning.

**Versioning in Amazon S3**

Amazon S3 can optionally store different versions of the same object. Have you ever been working on a document for hours and suddenly made a mistake where you deleted all of the content in the document, or have you made a big mistake and wanted to go back to a previous version? Many editors, such as Microsoft Word, offer the ability to undo changes and recover from some of these mistakes. However, once you save, close, and open the document again, you may not be able to undo any changes.

What if you have a document where multiple people make revisions, and you want to keep track of who made what changes?

Amazon S3 offers versioning capabilities that can assist with these use cases. So, what is versioning? Simply put, versioning is the ability to keep incremental copies. For example, if you store an important proposal document in S3, the first version of the document may have the initial architecture and statement of work, and the subsequent version may have evolved to looking at future architecture, which increases the scope of the work. Now if you want to compare these two versions, it is easy to view and recover the previous version, which has the original work statement.

As you can imagine, keeping multiple versions of the same document can get expensive if there are many changes. This is especially true if you have a high volume of documents. To reduce costs, we can implement a lifecycle policy where older versions are purged or moved to a cheaper storage option such as Amazon S3 Glacier.

The exact logic of implementing the lifecycle policy will depend on your requirements. But some possibilities are to set up your policy based on document age, the number of versions, or some other criteria. By the way, lifecycle policies are not limited to just older versions of a document. They can also be used for any document that persists in Amazon S3.

Due to S3 high availability, durability, and unlimited scalability, enterprises often use this to host critical workloads that may need backing up to other regions. Let’s learn about S3 Multi-destination replication, which allows users to distribute multiple copies to different environments.

**Amazon S3 Multi-Destination Replication**

Amazon S3 Multi-Destination Replication is a feature of Amazon S3 that allows users to replicate objects across multiple Amazon S3 buckets or AWS accounts. With Multi-Destination Replication, you can define a replication rule that specifies which objects should be replicated and where they should be replicated to.

To use Multi-Destination Replication, you first need to create a replication rule in the Amazon S3 console. This rule defines the source bucket where objects are stored, the destination buckets where objects should be replicated, and the prefixes or tags that identify which objects should be replicated.

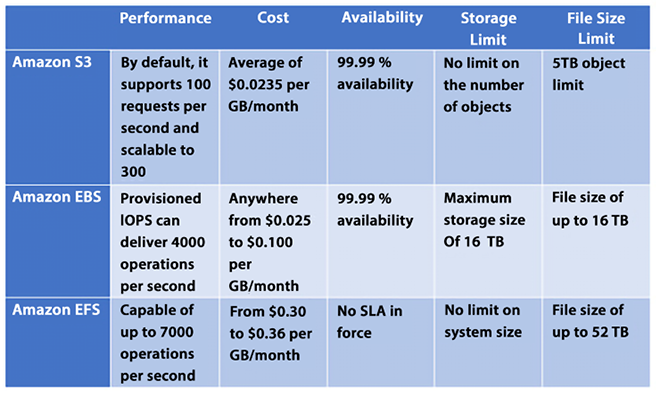
Once the replication rule is created, Amazon S3 will automatically replicate objects that match the specified criteria to the destination buckets. This replication is performed asynchronously, so the source and destination buckets do not have to be online at the same time.

S3 Multi-Destination Replication is a useful feature for users who need to replicate objects across multiple S3 buckets or AWS accounts. It allows you to easily replicate objects and can be used to improve the availability and durability of your data. To learn more, visit <https://docs.aws.amazon.com/AmazonS3/latest/userguide/replication-metrics.html>.

Now that we have covered Amazon Elastic Block Storage, Amazon Elastic File System, and Amazon S3, we will spend some time understanding the difference between the services and when it’s appropriate to use one versus the other.

**Choosing the right cloud storage type**

So far, you have learned about three different kinds of cloud storage in this chapter. First, Amazon EBS stores data in blocks; you can also use this as SAN in the cloud. Second, Amazon EFS is cloud file storage that is a kind of NAS in the cloud. Finally, Amazon S3 stores data as objects. So now that we covered all these storage type services, the obvious question is which one is better to use. The following table should help you to decide what service is best for your use case:



*Figure 5.3: Choosing the service based on your use case*

EBS volume is always attached to a single EC2 instance so when you need high-performance, persistent storage, always use EBS volume. If you need shared file storage between multiple EC2 instances, then you want to use EFS. S3 is your choice to store any amount of data in any format that you want to use for big data analytics, backups, and even large volume content for your application.

As you are going to use S3 a lot for your day-to-day large volume (from GBs to PBs) of data storage needs, let’s learn some best practices to manage S3.

**Exploring Amazon S3 best practices**

Amazon S3 is one of the simplest services in AWS, and at the same time, it is one of the most powerful and scalable services. We can easily scale our Amazon S3 applications to process thousands of requests per second while uploading and retrieving files. This scalability can be achieved “out of the box” without needing to provision any resources or servers.

Some customers in AWS are already leveraging Amazon S3 to host petabyte-scale data lakes and other applications storing billions of objects and performing billions of requests. These applications can upload and retrieve multiple terabytes of data per second with little optimization.

Other customers with low latency requirements have used Amazon S3 and other Amazon file storage services to achieve consistent low latency for small objects. Being able to retrieve this kind of object in 100 to 200 milliseconds is not uncommon.

For bigger objects, it is possible to achieve similar low latency responses for the *first byte* received from these objects. As you can imagine, the retrieval time to receive the complete file for bigger objects will be directly proportional to object size.

**Enhancing Amazon S3 performance**

One simple way to enhance performance is to know where most of your users are located and where your Amazon S3 bucket is located. Amazon S3 buckets need to be unique globally, but files will be stored in a given AWS Region. When you architect your solution, this is considered in your design. It will help reduce the time it takes to transfer files and minimize data transfer costs.

One more way to scale Amazon S3 is to scale S3 connections horizontally. Amazon S3 allows you to make many connections to any given bucket and access thousands of files per second. Highly scalable performance can be achieved by issuing multiple concurrent requests. Amazon S3 is designed to support high levels of performance and can handle a large number of requests per second. While the exact number of requests per second that the service can handle will depend on a variety of factors, it is capable of supporting at least 3,500 requests per second to add data and 5,500 requests per second to retrieve data. You can think of Amazon S3 as a highly distributed system and not just a single endpoint with only one server to support the workloads.

There are a number of ways that users can enhance the performance of Amazon S3, including the following:

* **Choose the right storage class**: S3 offers several different storage classes, each of which is optimized for different use cases. Choosing the right storage class for your data can help to improve performance and reduce latency.
* **Use caching**: S3 uses caching techniques to improve the performance of frequently accessed data. Enabling caching for your data can help to increase the number of requests per second that the service can handle and reduce the time it takes to retrieve data.
* **Use object partitioning**: S3 uses a partitioning scheme to distribute data across multiple servers and storage devices, which can help to increase the number of requests per second that the service can handle.
* **Use regional buckets**: S3 allows you to store data in regional buckets, which can help to reduce latency and improve performance by storing data closer to users.

Overall, there are many ways that users can enhance the performance of Amazon S3 and improve the speed and reliability of their data storage and retrieval. By choosing the right storage class, using caching, object partitioning, and regional buckets, users can maximize the performance of Amazon S3 for their specific use case.

We mentioned in the previous section that Amazon S3 files could be retrieved with sub-second performance. However, suppose this level of performance is not enough, and you are looking to achieve single-digit millisecond performance. In that case, you can use Amazon CloudFront to store data closer to your user base to achieve even higher performance, or Amazon ElastiCache to cache the data in memory and reduce data load time.

**Amazon CloudFront**

You learned about CloudFront in *Chapter 4*, *Networking in AWS*. Amazon CloudFront is a **Content Delivery Network** (**CDN**) that can cache content stored in Amazon S3 and distribute it across dispersed geographic regions with thousands of **Points of Presence** (**PoP**) worldwide. Amazon CloudFront enables these objects to be cached close to the people using these resources.

**Amazon ElastiCache**

Amazon ElastiCache is a managed AWS service that enables you to store objects in memory instead of storing them on disk. Behind the scenes and transparently, Amazon ElastiCache will provision Amazon EC2 instances that will cache objects in the instance’s memory. Doing so will reduce latency when retrieving objects by orders of magnitude. Using Amazon ElastiCache does require subtly changing application logic.

First, when you want certain objects to be cached instead of stored on a disk, you need to specify that ElastiCache should be the storage medium. And when you retrieve objects, you should check the cache in ElastiCache to see whether the object you are trying to retrieve has been cached and, if they haven’t, only then check the database to get the uncached object.

While CloudFront provides caching services at the edge for data distribution, ElastiCache handles data at the application level to load data faster when you request it for business logic calculations or displays it to the end user through the application web layer. Amazon ElastiCache comes in two flavors: Redis (persistent cache) and Memcached (fast indexing). You will learn about these in more detail in *Chapter 7*, *Selecting the Right Database Service*.

**Amazon S3 Transfer Acceleration**

Yet another way to achieve single-digit millisecond responses is to use Amazon S3 Transfer Acceleration. This AWS service uses CloudFront edge locations to accelerate data transport over long distances. Amazon S3 Transfer Acceleration is ideally suited to transfer a large amount of data (gigabytes or terabytes) that needs to be shared across AWS Regions.

It would help if you considered using Amazon S3 Transfer Acceleration in the following cases:

* Application requirements call for the need to upload files to a central location for many places around the globe.
* There is a need to regularly transfer hundreds of gigabytes or terabytes worth of data across AWS Regions.
* The available bandwidth is not being fully utilized and leveraged when uploading to Amazon S3.

The benefits of Amazon S3 Transfer Acceleration are as follows:

* It will allow you to transfer files faster and more consistently over long distances.
* It can reduce network variability usage.
* It can shorten the distance traveled to upload files to S3.
* It will enable you to maximize bandwidth utilization.

One critical consideration when using Amazon S3 is ensuring that the data stored is accessible only by parties that need to access the data. Everyone else should be locked out. Let’s learn about AWS’s capabilities to assist in data protection and data security.

**Choosing the right S3 bucket/prefix naming convention**

S3 bucket names must be unique, meaning that no two buckets can have the same name. Once a bucket is deleted, its name can potentially be reused, although there are some exceptions to this and it may take some time before the name becomes available again. It is therefore recommended to avoid deleting a bucket if you want to reuse its name.

To benefit from new features and operational improvements, as well as virtual host-style access to buckets, it is recommended to use bucket names that comply with DNS naming conventions, which are enforced in all regions except US East. When using the AWS Management Console, bucket names must be compliant with DNS naming conventions in all regions.

Here are some best practices for differentiating between file and bucket key names when uploading a large number of objects to S3:

* Avoid using the same key name for different objects. Each object should have a unique key name to avoid overwriting existing objects with the same name.
* Use a consistent naming convention for object key names. This will make it easier to locate and manage objects later on.
* Consider using a hierarchical naming structure for your objects. For example, you could use a directory structure in your object key names to help organize objects into logical groups.
* Use a delimiter, such as a slash (/), to separate the directory structure in your object key names. This will help you to easily navigate the structure and make it easier to organize and manage your objects.
* Avoid using special characters in your key names, as this can cause compatibility issues with different systems and applications.
* Choose key names that are descriptive and easy to understand. This will make it easier for others to navigate and understand your objects.

To ensure workload efficiency, it is recommended to avoid sequential key names (or add a random prefix) if a workload is expected to exceed 100 requests per second. This can help to evenly distribute key names across multiple index partitions, thereby improving the workload distribution and overall system performance.

All of the above best practices apply to S3 bucket naming. Amazon S3 stores bucket names as part of key names in its index.

**Protecting your data in Amazon S3**

Amazon makes it extremely easy to provision an Amazon S3 bucket and quickly allows you to distribute the data worldwide by simply providing a URL pointing to files in the bucket. The good news is that this is so easy to do. The bad news is that this is so easy to do.

There have been many documented cases of sensitive data lying around in publicly accessible endpoints in S3 and other people accessing this data. “Leaky” Amazon S3 buckets are a perfect example of how the shared responsibility security model works when using AWS.

AWS provides a fantastic array of security services and protocols, but data can still be exposed if not used correctly, and breaches can occur. By default, each bucket is private but you need to configure your Amazon S3 buckets and object security with the principle of least privilege when starting to provide access.

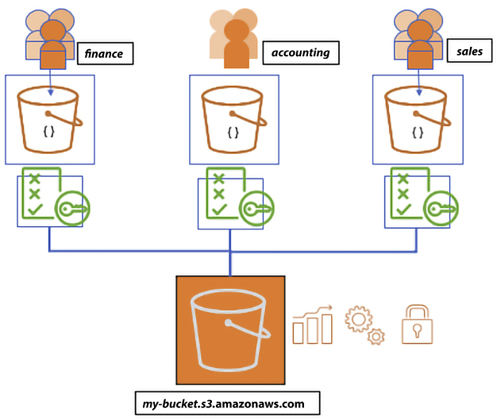
The good news is that as easy as it is to make the bucket public and leave it open to the world, it is almost as simple to lock it down and restrict access only to the required individual and services.

Some of the features that Amazon S3 provides to restrict access to the correct users are as follows.

**Blocking Amazon S3 public access to buckets and objects whenever possible**

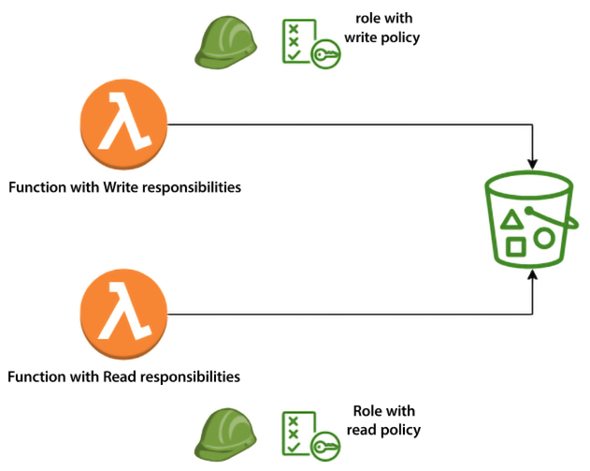
Starting with the principle of least privilege, AWS blocks public access to any newly created S3 bucket by default. Amazon S3 provides a bucket policy where you can define who can access this S3 resource. **Amazon S3 Access Control Lists** (**ACLs**) provide granular control at the object level. You can learn more about S3 ACL by visiting the AWS user document here - <https://docs.aws.amazon.com/AmazonS3/latest/userguide/acl-overview.html>.

Further, AWS provides S3 Access Points, which are simplified management for shared bucket access by many teams. For example, a data lake bucket where you want to store all your organization data as a single source can be accessed by the finance, accounting, and sales teams, and as shown below, you can use S3 Access Points to restrict access for each team to have access to their data only based on a defined prefix.



*Figure 5.4: Amazon S3 Access Points*

As shown in the diagram above, users are segmented into distinct groups and each group is given their own S3 access point through the specific policies applied to the group. This helps to manage access policies centrally across multiple users. The following is an example for segregation of Read and Write access in a bucket:

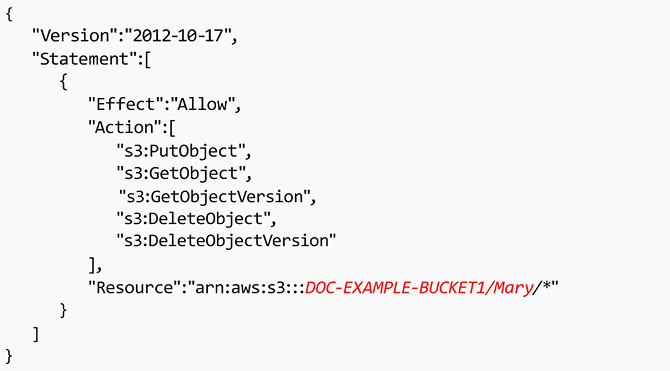


*Figure 5.5: Amazon S3 access policy segregation*

Leveraging Amazon S3 to block public access, Amazon S3 bucket administrators can configure a way to control access in a centralized manner and limit public access to Amazon S3 buckets and objects. This feature can deny public access regardless of how objects and buckets are created in Amazon S3.

**Avoiding wildcards in policy files**

Policy files allow a powerful syntax where you can use a wildcard character (**\***) to specify policies. Even though wildcards are permitted, they should be avoided whenever possible, and instead, names should be spelled out explicitly to name resources, principles, and others. The following is a sample policy with no wildcards:



*Figure 5.6: Sample policy with no wildcards*

The same wildcard rule applies to Amazon S3 bucket **Access Control Lists** (**ACLs**). ACLs are files that can be used to deliver read, write, or full access to users, and if wildcards are used, they can potentially leave the bucket open to the world.

**Leveraging the S3 API**

Like other AWS services, Amazon S3 provides hundreds of APIs that you can use through the AWS **CLI** (**Command-Line Interface**) or call the API from your application code. For example, the ListBuckets API can be used to scan Amazon S3 buckets in a given AWS account. GetBucketAcl returns the ACL of a bucket and GetBucketWebsite returns the website configuration for a bucket, and GetBucketPolicy commands can monitor whether buckets are compliant and whether the access controls, policies, and configuration are properly set up to only allow access to authorized personnel. You can refer to AWS user docs to get the list of all the APIs and how to use them through the CLI or application; please find the user docs here: <https://docs.aws.amazon.com/AmazonS3/latest/API/Welcome.html>.

**Leveraging IAM Access Analyzer to inspect S3**

AWS **Identity and Access Management** (**IAM**) Access Analyzer for S3 can generate comprehensive findings if your resource policies grant public or cross-account access. It continuously identifies resources with overly broad permissions across your entire AWS organization. It resolves results by updating policies to protect your resources from unintended access before it occurs or archives findings for intended access.

IAM Access Analyzer continuously monitors for new or updated policies. It analyzes permissions granted using policies for your Amazon S3 buckets, AWS **Key Management Service** (**KMS**) keys, AWS IAM roles, and AWS Lambda functions, and generates detailed findings about who has access to what resources from outside your AWS organization or AWS account.

If the level of resource access is not intended, modify the resource policy to further restrict or expand access to the resource. If the access level is correct, you can archive the finding. IAM Access Analyzer provides comprehensive results that can be accessed through the AWS IAM, Amazon S3, and AWS Security Hub consoles and APIs. This service delivers detailed findings, allowing users to easily view and analyze access-related issues and take appropriate action to address them.

**Enabling AWS Config**

Another way to verify your security configuration is to deploy a continuous monitoring system using s3-bucket-public-read-prohibited and s3-bucket-public-write-prohibited and manage the configuration using AWS Config rules.

AWS Config is a service that can monitor and evaluate the way resources are configured in your AWS setup. AWS Config continuously audits the environment to comply with pre-established desired configurations. If a configuration deviates from the expected standards, alerts can be generated, and warnings can be issued. As you can imagine, it doesn’t apply to just Amazon S3 but all AWS services. You will learn more about AWS Config in *Chapter 8*, *Best Practices for Application Security, Identity, and Compliance*.

AWS greatly simplifies the monitoring of your environment, enhances troubleshooting, and ensures that established standards are followed.

**Implementing S3 Object Lock to secure resources**

S3 Object Lock allows the storage of objects with **Write Once Read Many** (**WORM**) models. S3 Object Lock assists in preventing the accidental or nefarious deletion of important information. For example, S3 Object Lock can be used to ensure the integrity of AWS CloudTrail logs.

**Implementing data at rest encryption**

AWS provides multiple encryption options for S3 as explained below:

* **Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)**: To use SSE-S3, you simply need to enable the feature for your S3 bucket and Amazon S3 will automatically encrypt and decrypt your data using keys that are managed by the service.

SSE-S3 uses 256-bit Advanced Encryption Standard (AES-256) encryption, which is one of the most secure encryption algorithms available. SSE-S3 is a convenient and secure way to encrypt data at rest in Amazon S3. It automatically encrypts and decrypts your data using keys that are managed by Amazon S3, providing a simple and secure way to protect your data.

* **Server-Side Encryption with AWS Key Management Service (SSE-KMS)**: SSE-KMS is similar to SSE-S3, but with some additional benefits and charges. AWS KMS is a service that is fully managed, and enables the creation and management of **Customer Master Keys** (**CMKs**), which are the keys used for encrypting your data. For using a CMK, there are different permissions available that provide extra security against unauthorized access to your Amazon S3 objects. Using the SSE-KMS option, an audit trail can be created that indicates when and by whom the CMK was used. The option also enables you to create and manage customer-managed CMKs or use AWS-managed CMKs that are exclusive to your service, your Region, and you.
* **Server-Side Encryption with Customer-Provided Keys (SSE-C)**: In the case of SSE-C, you are responsible for managing the encryption keys while Amazon S3 is responsible for encryption during the write to disk process and the decryption process when accessing your objects.
* **Client-side encryption**: You can encrypt data before sending it to Amazon S3. You have the freedom to choose your own encryption method but you are responsible for managing the encryption key.

**Enabling data-in-transit encryption**

Amazon S3 supports HTTPS (TLS) to prevent attacks to nefariously access or modify the traffic between your users and the Amazon S3 buckets. It is highly recommended to modify your Amazon S3 bucket policies only to permit encrypted connections that use the HTTPS (TLS) protocol.

It is also recommended to implement a rule in AWS Config that enables continuous monitoring controls using **s3-bucket-SSL-requests-only**.

**Turning on Amazon S3 server access logging**

By default, access is not logged. This ensures that you are not charged for the storage space these logs will take. However, it’s relatively easy to turn on logging. These logs will give you a detailed record of any traffic in Amazon S3. These access logs will help determine who accessed your buckets and when they were accessed.

This will help not only from a security perspective but also to assess traffic patterns and help you to control your costs. You can also enable AWS CloudTrail to log all activity related to S3.

S3 server access logging allows users to log requests made to their S3 bucket. With server access logging, you can capture detailed information about every request made to their S3 bucket, including the requestor’s IP address, the request type, the response status, and the time of the request.

To turn on server access logging for your S3 bucket, you first need to create a target bucket where the logs will be stored. This bucket can be in the same AWS account as your source bucket or in a different account.

Once you have created the target bucket, you can enable server access logging for your source bucket. As soon as server access logging is enabled, Amazon S3 will automatically log requests made to your bucket and store the logs in the target bucket. You can view the logs by accessing the target bucket in the Amazon S3 console, or by using the Amazon S3 API.

**Considering the use of Amazon Macie with Amazon S3**

Amazon Macie leverages the power of machine learning to automatically ensure that sensitive information is not mishandled when using Amazon S3. Amazon Macie can locate and discern sensitive data in Amazon S3 and provide data classification. Macie can recognize **Personally Identifiable Information** (**PII**), intellectual property, and similar sensitive information. Amazon Macie has instrumentation panels, reports, and warnings to show how data is used.

**Implementing monitoring leveraging AWS monitoring services**

Monitoring is a critical component of any computing solution. AWS provides a variety of services to consistently and reliably monitor Amazon S3, such as CloudWatch which offers a variety of metrics for Amazon S3 including the number of put, get, and delete requests.

**Using VPC endpoints to access Amazon S3 whenever possible**

Using **Virtual Private Cloud** (**VPC**) endpoints with Amazon S3 enables the use of Amazon S3 without traversing the internet and minimizing risk. A VPC endpoint for Amazon S3 is an artifact within an Amazon VPC that only allows connections from Amazon S3. To allow access from a given Amazon VPC endpoint, an Amazon S3 bucket policy can be used.

The following is the VPC endpoint policy:



*Figure 5.7: VPC endpoint policy*

Below is the S3 bucket policy to accept a request from the VPC endpoint:



*Figure 5.8: S3 bucket policy*

VPC endpoints for Amazon S3 use two methods to control access to your Amazon S3 data:

* Controlling the requests made to access a given VPC endpoint
* Controlling the VPCs or VPC endpoints that can make requests to a given S3 bucket by taking advantage of S3 bucket policies

Data exfiltration can be prevented by leveraging a VPC without an internet gateway.

To learn more, please check the following link: <https://docs.aws.amazon.com/AmazonS3/latest/dev/example-bucket-policies-vpc-endpoint.html>.

**Leveraging Amazon S3 cross-region replication**

By default, Amazon S3 persists data in more than one AZ. Still, there may be other reasons that require you to provide an even higher level of redundancy—for example, storing the data in different continents due to compliance requirements or minimizing the data access latency for the team situated in another region. **Cross-Region Replication** (**CRR**) is the storage of data across AWS Regions to satisfy those types of requirements. CRR enables asynchronous replicating objects across AWS Regions in multiple buckets.

In this section, you learned about some of the recommended best practices when implementing Amazon S3 as part of any solution, including how to achieve the best performance, minimize sticker shock at the end of the month with your monthly charges by monitoring usage, and ensure that only “need to know” users access data in your Amazon S3 buckets.

**Amazon S3 cost optimization**

You should configure your Amazon S3 to manage your items and keep them cost-effectively stored throughout their lifecycle. You can use an S3 Lifecycle configuration to make S3 move data from S3 Standard to S3 Glacier after it hasn’t been accessed for a certain period of time. This can help you reduce your storage costs by storing data in the most cost-effective storage class. Additionally, you can use the following to reduce costs further:

* **Data transitions**: Using Amazon S3 storage classes, these define when objects transfer from one storage class to another. For example, one year later, archive it to the S3 Glacier storage class.
* **Data expirations**: Specify when objects will expire. On your behalf, Amazon S3 deletes expired objects. When you choose to expire items, the lifetime expiration costs vary. In case you transfer intermittent logs to a bucket, your application might require them for a week or a month.

After that, you might need to erase them. Some records are habitually written for a restricted period of time. After that, they are rarely accessed.

* **Archiving**: At some point you may no longer be accessing data, but your organization might require you to archive it for a particular period for administrative compliance. S3 Glacier is a proper choice for enterprises that only need to reference a group of data once or twice a year, or for backup purposes. Glacier is Amazon’s most affordable storage class. When compared to other Amazon storage offerings, this allows an organization to store large amounts of data at a much lower cost. S3 Standard is suitable for frequently accessed data, while S3 Glacier is better suited for infrequently accessed data that can tolerate longer retrieval times. Choosing the right storage class can help you save money by only paying for the level of storage and retrieval performance that you need.
* **Automated cost saving**: S3 Intelligent-Tiering automatically stores objects in three access tiers:
  + A Frequent Access tier if you want access data more often.
  + An Infrequent Access tier which has 40% lower cost than the Frequent Access tier (object is not accessed for 30 consecutive days).
  + A new Archive Instant Access tier with 68% lower cost than the Infrequent Access tier (object is not accessed for 90 consecutive days).

S3 Intelligent-Tiering monitors access patterns and moves objects that have not been accessed for 30 consecutive days to the Infrequent Access tier, and now, after 90 days of no access, to the new Archive Instant Access tier.

Let’s say you have an S3 bucket that contains a large number of objects, some of which are accessed frequently and some of which are accessed infrequently. If you were to use the S3 Standard storage class to store all of these objects, you would pay the same price for all of them, regardless of their access patterns.

With S3 Intelligent-Tiering, however, you can take advantage of the cost savings offered by the infrequent access tier for the less frequently accessed objects. For example, let’s say you have a set of log files that are accessed frequently during the first 30 days after they are created, but then only occasionally after that. With S3 Intelligent-Tiering, these log files would be automatically moved to the infrequent access tier after 30 days of inactivity. This would result in cost savings compared to storing them in the frequent access tier for the entire time.

The movement of objects between tiers in S3 Intelligent-Tiering is fully automated and does not require any management or intervention from the user. You are charged a small monitoring and automation fee in addition to the storage and data transfer fees associated with the storage class.

Note that by opting into asynchronous, archive capabilities for objects that are rarely accessed, you can realize storage cost savings of up to 95%, with the lowest storage cost in the cloud.

You can use S3 Transfer Acceleration to reduce data transfer costs. S3 Transfer Acceleration allows you to transfer large amounts of data to S3 over long distances more quickly and cheaply than using the internet alone. This can help you save on data transfer costs, especially if you transfer a lot of data between regions or between AWS and your on-premises data centers.

Using S3 batch operations to reduce the number of requests made to the service. S3 batch operations allow you to perform multiple operations on your data in a single request, which can help you reduce the number of requests you make to the service and save on request fees.

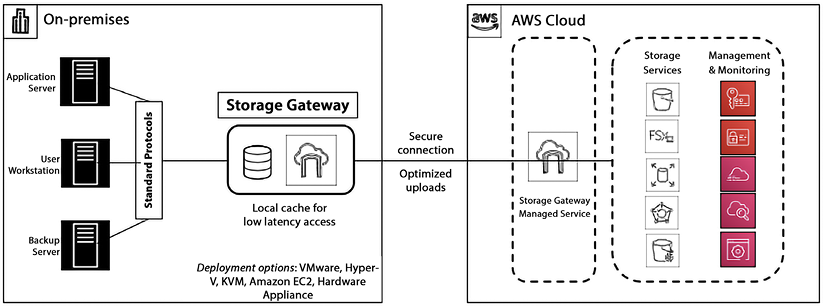
Many application workloads on-premises today either can’t be moved to the cloud or are challenging to move. Some of the more common application examples include genomic sequencing, media rendering, medical imaging, autonomous vehicle data, seismic data, manufacturing, etc. AWS provides Storage Gateway to connect those applications to the cloud.

**Building hybrid storage with AWS Storage Gateway**

While working on cloud migration, some applications will not be so simple to move to the cloud. Those apps might need to stay on-premises for performance reasons or compliance reasons, or they may be too complex to move into the cloud quickly. Some apps may need to remain on-premises indefinitely such as mainframe applications or legacy applications that need to meet licensing requirements. To address these use cases, you need to explore hybrid cloud storage solutions that provide ready access for on-premises apps to data stored in AWS.

**AWS Storage Gateway**

AWS Storage Gateway acts as a bridge to provide access to almost unlimited cloud storage by connecting applications running on-premises to Amazon storage. As shown in the diagram below, Storage Gateway allows customers to connect to and use key cloud storage services such as Amazon S3, Amazon S3 Glacier, Amazon FSx for Windows File Server, and Amazon EBS. Additionally, Storage Gateway integrates with AWS services such as AWS KMS, AWS IAM, AWS CloudTrail, and AWS CloudWatch.

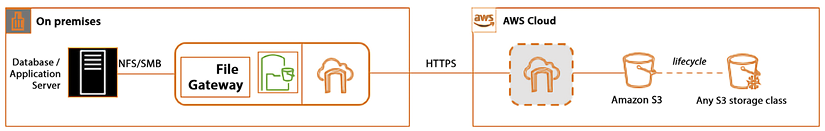


*Figure 5.9: AWS Storage Gateway*

Storage Gateway quickly deploys on-premises as a preconfigured hardware appliance. There is also a virtual machine option that supports all the major hypervisors. Storage Gateway provides a local cache to enable access to frequently accessed data with low-latency access. Storage Gateway supports access via standard storage protocols (NFS, SMB, and iSCSI VTL), so no changes to customers’ applications are required. There are four types of Storage Gateway.

**Amazon S3 File Gateway**

Amazon S3 File Gateway is a service that allows you to store and retrieve files from Amazon S3 using the file protocol (i.e., **Network File System** (**NFS**) and **Server Message Block** (**SMB**)). This means that you can use S3 File Gateway as a file server, allowing you to access your S3 objects as if they were files on a local file system. This can be useful for applications that require access to files stored in S3, but don’t support object storage directly. You can see the data flow in the diagram below.



*Figure 5.10: Amazon S3 File Gateway*

To use S3 File Gateway, you simply create a file share and configure it to store files in your S3 bucket. You can then access the file share using the file protocol, either from within your Amazon VPC or over the internet. This allows you to easily integrate S3 File Gateway with your pre-existing applications and workflows.

Overall, S3 File Gateway is a useful service for applications that require file storage but want to take advantage of the scalability, durability, and cost-effectiveness of S3.

S3 File Gateway supports various features, such as versioning, data deduplication, and data tiering. It also integrates with other AWS services, such as AWS Backup, AWS IAM, and AWS **Key Management Service** (**KMS**), allowing you to use these services with your S3 objects.

**Amazon FSx File Gateway**

Amazon FSx File Gateway is a hybrid cloud storage solution that provides on-premises applications access to virtually unlimited cloud storage. It allows you to use your existing file servers and **Network-Attached Storage** (**NAS**) devices as a seamless interface to Amazon S3, Amazon FSx for Lustre, and Amazon FSx for Windows File Server. Here’s an example of how Amazon FSx File Gateway works:

Suppose you have an on-premises file server that you use to store important files and data. As your storage needs grow, you begin to run out of space on your local file server. Instead of purchasing additional storage or upgrading your hardware, you can use Amazon FSx File Gateway to seamlessly extend your storage to the cloud.

First, you deploy a virtual machine on your on-premises infrastructure and install the Amazon FSx File Gateway software on it. This virtual machine serves as a gateway between your on-premises file server and Amazon S3 or Amazon FSx. Next, you create an Amazon S3 bucket or an Amazon FSx file system to store your files in the cloud. You can choose different S3 storage classes and lifecycle policies to manage the cost of your cloud storage. Then, you create a file share on the Amazon FSx File Gateway virtual machine and connect it to your on-premises file server using standard protocols such as **Server Message Block** (**SMB**) or **Network File System** (**NFS**). You can use your existing file server permissions and Active Directory to manage access to the files.

Finally, when you save a file on your on-premises file server, it is automatically backed up to the cloud using the Amazon FSx File Gateway. You can also access the files in the cloud directly from the file server without any additional steps.

Amazon FSx File Gateway also provides features such as caching, multi-protocol access, and file-level restore, making it an ideal solution for backup and disaster recovery, content distribution, and data archiving. It allows you to use your existing on-premises file servers to store and access files on the cloud, providing virtually unlimited storage capacity without the need for additional hardware or complex software configurations.

**Tape Gateway**

Amazon Tape Gateway is a service that allows you to store data on tapes using the tape protocol (i.e., **Linear Tape-Open** (**LTO**) and **Virtual Tape Library** (**VTL**)). This means that you can use Tape Gateway as a tape library, allowing you to access your data as if it were stored on tapes in a local tape library. This can be useful for applications that require access to data stored on tapes, but don’t support tapes directly.

Tape Gateway supports various features, such as data deduplication, data tiering, and encryption. It also integrates with other AWS services, such as AWS Storage Gateway, AWS Backup, and AWS KMS, allowing you to use these services with your data on tapes.

To use Tape Gateway, you simply create a tape virtual device and configure it to store data on tapes in your tape library. You can then access the tape virtual device using the tape protocol, either from within your VPC or over the internet. This allows you to easily integrate Tape Gateway with your existing applications and workflows. Overall, Tape Gateway is a useful service for applications that require tape storage but want to take advantage of the scalability and durability of AWS.

**Volume Gateway**

Amazon Volume Gateway is a service that allows you to store data on cloud-backed storage volumes using the iSCSI protocol. This means that you can use Volume Gateway as a storage device, allowing you to access your data as if it were stored on a local storage volume. This can be useful for applications that require access to data stored on a storage volume, but don’t support cloud storage directly.

Volume Gateway supports two storage modes: cached and stored. In cached mode, data is stored on your local storage volume and asynchronously backed up to Amazon S3, allowing you to access your most frequently accessed data quickly while still providing long-term durability. In stored mode, data is directly stored on Amazon S3, allowing you to store large amounts of data without the need for local storage.

To use Volume Gateway, you simply create a storage volume and attach it to your on-premises or Amazon EC2 instance. You can then access the storage volume using the iSCSI protocol, either from within your Amazon VPC or over the internet. This allows you to easily integrate Volume Gateway with your existing applications and workflows. Overall, Volume Gateway is a useful service for applications that require storage volumes but want to take advantage of the scalability and durability of AWS.

Storage Gateway offers customers the advantages of hybrid cloud storage by providing a seamless migration path to the cloud. Customers experience a fast deployment, which enables them to leverage the agility and scale of the cloud quickly. Storage Gateway doesn’t require any application changes. It easily integrates with standard storage protocols on-prem.

Finally, Storage Gateway is managed centrally via the AWS Console. It integrates with various AWS services like CloudWatch, CloudTrail, and Identity Access Management to provide customers visibility and control over the whole solution. You often want to back up your data for various reasons, like disaster recovery. In such cases, you need an easy option to back up your data facilitated by AWS using AWS Backup.

**AWS Backup**

AWS Backup is a service that centralizes backup management and enables a straightforward and economical means of backing up application data across multiple AWS services to help customers comply with their business continuity and backup requirements. It automates backup scheduling and retention management, and it provides a centralized way for configuring and auditing the resources that require backup. Additionally, it keeps an eye on backup activity and alerts you in case of any issues. AWS Backup integrates with CloudTrail and AWS Organizations for governance and management, giving customers many options to help meet their recovery, restoration, and compliance needs.

AWS Backup enables centralized configuration and management of backups for various AWS resources including Amazon EC2 instances, Amazon EBS volumes, Amazon **Relational Database Service** databases, Amazon DynamoDB tables, Amazon EFS file systems, and other resources. You will learn about all of the database services we have just mentioned in *Chapter 7*, *Selecting the Right Database Service*. Some of the use cases where you may want to use AWS Backup are:

* Compliance and disaster recovery
* Unifying backup solutions to avoid the complexity of cloud backups being done by different groups
* Creating audit and compliance alerts, reports, and dashboards across all backups

To use AWS Backup, you simply create a backup plan and specify the AWS resources you want to back up. AWS Backup will then automatically create backups according to your specified schedule and store them in your specified storage location. You can then restore your backups as needed, either to the original location or to a new location. This allows you to easily manage your backups and recover from data loss in the event of a disaster. Overall, AWS Backup is a useful service for businesses that want to ensure the durability and availability of their critical data on AWS.

In this section, you learned about various Storage Gateway options to build a hybrid cloud by storing data into AWS cloud and AWS Backup to provide a cloud-native option for data backup.

**Summary**

In this chapter, you learned about **Storage Area Network**(**SAN**) in the cloud with Amazon EBS. You learned about various EBS options and how to choose the right EBS volume per your workload. You further learned about **Network Attached Storage**(**NAS**) in the cloud with Amazon EFS and file system-specific workloads with Amazon FSx.

With the ever-increasing amount of data, you need scalable storage to store petabytes of data, and AWS provides Amazon S3 to fulfill that need. You learned about various tiers of Amazon S3, including S3 Standard, Intelligent Tiering, Infrequent Access-IA, One Zone-IA, and S3 Glacier. You further learned about S3 versioning to save a copy of your file and build multi-destination replication.

Later in the chapter, you learned about Amazon S3 best practices and optimized your S3 storage for performance, cost, and security. Finally, you learned about building a hybrid cloud with AWS Storage Gateway and a cloud-native backup option with AWS Backup.

In the next chapter, you will learn how to harness the power of the cloud to create powerful applications, and we will do a deep dive into another important AWS service—Amazon EC2.