

The Guide to AWS Cost Optimization



Introduction

This guide contains a series of advanced cost-optimization techniques that can lower your AWS bill without compromising application performance or redundancy preparedness for disaster prevention. These articles are written for AWS Administrators and cloud engineering managers wanting to gain a deeper understanding of lesser-known aspects of AWS pricing.

Have questions about other cost-optimization methods or AWS Services? Send us a request! CloudBolt adds chapters to this guide regularly as best practices and AWS products evolve.



The AWS Cost Optimization Puzzle

The Basics of AWS Cost Optimization

At its most basic level, cloud cost optimization is a method of practice that should be broken down into the following cycle:

1. **Build asset awareness** through inventory analysis, tagging, and tracking. Make sure you understand not only what resources you have, but also their relationships with one another and the applications they support.
2. **Collect key metric data** to set benchmarks for month-to-month and year-to-year trend analysis. This enables you to forecast budgetary needs confidently due to the cataloging achieved in step 1.
3. **Refresh your knowledge** of available services, resources, and discount programs.
4. **Execute a data-driven purchasing strategy** that applies all previous steps. This plan should factor in right-sizing resources, analyzing existing commitments for utilization and efficacy, upgrading to relevant instance generations, and validating new commitments against a 1- or 3- year horizon.

The following sections focus on phase 3 of the cost-optimization cycle and assume that you already have a cataloged infrastructure with historical cost and usage data.

Chapters

There's a lot of ground to cover regarding AWS services, pricing models, pitfalls, and best practices. Feel free to jump around to chapters you feel more comfortable tackling first. Remember, every organization has different needs. You can apply the cost-optimization cycle to different dimensions (data-flow routing, storage cleanup, commitment analysis, etc.) at different times to prevent concentrated (read: stressful) resource-management cliffs.

Chapter 1: AWS EBS Pricing

Discover all the ways you can improve your storage costs with our top 12 most important EBS optimization tips.

Chapter 2: AWS Data Transfer Pricing

We've simplified the rates and stipulations on data transfers across all of the most popular AWS products so that you can architect your data-flow with confidence.

Chapter 3: AWS Disaster Recovery Pricing

Understand the cost implications of the most common data replication strategies and gain insight into determining which might best suit your business needs.

Chapter 4: AWS Savings Plans

Round out your savings strategy with the latest AWS discount offering and see how it differs from standard AWS Reserved Instances.

Chapter 5: AWS Cost Allocation Tags

Learn about types of tags, naming conventions, and best practices, and how to use tags to organize and optimize your spending.

Chapter 6: EC2 Spot Instances

Reduce your AWS bill by mastering spot fleets, fleet requests, and more in our guide to spot instances.

Chapter 7: AWS Bill Analysis

Get to know all of the distinct use cases for AWS billing tools, what categories they fall in, and some best practices for using these tools.

Chapter 8: AWS Sizing Tools

Use different AWS sizing tools for your resources based on the data, goals, approach that best fits your projects.

Chapter 1: AWS EBS Pricing

Guide To AWS EBS Pricing

Amazon Elastic Block Storage (EBS) is an easy to use, high-performance block storage service designed to work with Amazon Elastic Compute Cloud (EC2). You can change volume types, tune performance, dynamically increase volume size and modify the provisioned IOPS capacity on live production environments, without disrupting your critical applications running on it. Multiple EBS volumes can be attached to a single instance if all resources are in the same Availability Zone. You can also use [Multi-Attach](#) to mount a volume to multiple instances at the same time.

In this article, we explain how EBS volumes are priced, highlight the characteristics of the general purpose SSD volume known as GP3, and present the industry best practices for controlling your EBS spending.

Types of EBS Volumes

Volume Type	EBS Provisioned IOPS SSD (io2)	EBS Provisioned IOPS SSD (io1)	EBS General Purpose SSD (gp3)	EBS General Purpose SSD (gp2)	Throughput Optimized HDD (st1)	Cold HDD (sc1)
Description	Highest performance and highest durability SSD volume designed for latency-sensitive transactional workloads	Highest performance SSD volume designed for latency-sensitive transactional workloads	Lowest cost SSD volume designed for various types of transactional workloads	General Purpose SSD volume that balances price performance for a wide variety of transactional workloads	Low cost HDD volume designed for frequently accessed, throughput intensive workloads	Lowest cost HDD volume designed for less frequently accessed workloads
Durability	100.00%	99.8% - 99.9% durability	99.8% - 99.9% durability	99.8% - 99.9% durability	99.8% - 99.9% durability	99.8% - 99.9% durability
Use Cases	I/O-intensive NoSQL and relational databases	I/O-intensive NoSQL and relational databases	Boot volumes, low-latency interactive apps, dev and test	Boot volumes, low-latency interactive apps, dev and test	Big data, data warehouses, log processing	Colder data requiring fewer scans per day
API Name	io2	io1	gp3	gp2	st1	sc1
Volume Size	4 GB - 16 TB	4 GB - 16 TB	1 GB - 16 TB	1 GB - 16 TB	500 GB - 16 TB	500 GB - 16 TB
Max IOPS/Volume	64,000	64,000	16,000	16,000	500	250
Max Throughput/Volume	1,000 MB/s	1,000 MB/s	1,000 MB/s	250 MB/s	500 MB/s	250 MB/s
Max IOPS/Instance	160,000	160,000	260,000	260,000	160,000	160,000
Max Throughput/Volume	4,750 MB/s	4,750 MB/s	7,500 MB/s	7,500 MB/s	4,750 MB/s	4,750 MB/s
Price	\$0.125/GB-month \$0.065/provisioned IOPS	\$0.125/GB-month \$0.065/provisioned IOPS	\$0.08/GB-month, 3,000 IOPS free and \$0.005/provisioned IOPS-month over 3,000, 125 MB/s free and \$0.04/provisioned MB/s-month over 125	\$0.10/GB-month	\$0.045/GB-month	\$0.025/GB-month
Dominant Performance Attribute	IOPS and volume durability	IOPS	IOPS	IOPS	MB/s	MB/s

How EBS pricing works

See the following table for a thorough breakdown of pricing for different EBS products.

Volume Type	Description	AWS EBS Type	Pricing
Free Tier	AWS Free Tier includes 30GB of Storage, 2 million I/Os, and 1GB of snapshot storage with Amazon Elastic Block Store (EBS)	For Certain volume types	Free
Amazon EBS Volumes	Pay only for what you use.	General Purpose SSD (gp3) Volumes	\$0.08/GB-month, 3,000 IOPS free and \$0.005/provisioned IOPS-month over 3,000, 125 MB/s free and \$0.04/provisioned MB/s-month over 125
Amazon EBS Volumes	Pay only for what you use.	General Purpose SSD (gp2) Volumes	\$0.10 per GB-month of provisioned storage
Amazon EBS Volumes	Pay only for what you use.	Provisioned IOPS SSD (io2) Volumes	\$0.125 per GB-month of provisioned storage AND \$0.065 per provisioned IOPS-month
Amazon EBS Volumes	Pay only for what you use.	Provisioned IOPS SSD (io1) Volumes	\$0.125 per GB-month of provisioned storage AND \$0.065 per provisioned IOPS-month
Amazon EBS Volumes	Pay only for what you use.	Throughput Optimized HDD (st1) Volumes	\$0.045 per GB-month of provisioned storage
Amazon EBS Volumes	Pay only for what you use.	Cold HDD (sc1) Volumes	\$0.025 per GB-month of provisioned storage

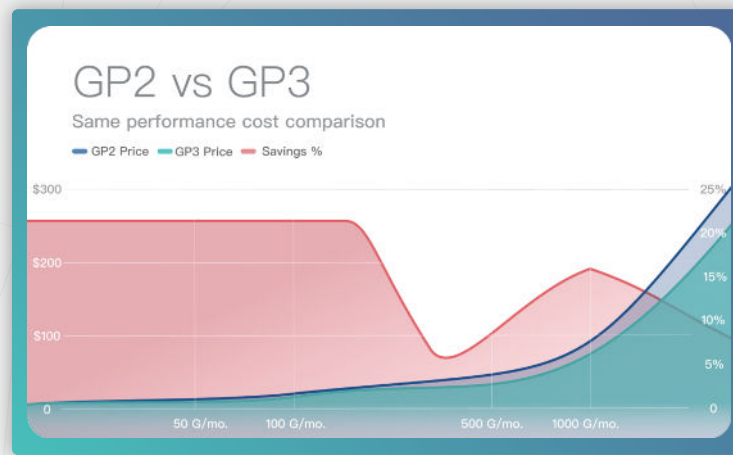
Amazon EBS Snapshots	Amazon EBS Snapshots are a point in time copy of your block data. EBS Snapshots are stored incrementally, which means you are billed only for the changed blocks stored.	EBS Snapshots	\$0.05 per GB-month of data stored
Amazon EBS Fast Snapshot Restore	Fast Snapshot Restore (FSR) allows you to promptly restore volumes. FSR is charged in Data Services Unit-Hours (DSU-Hours) until you disable FSR on the snapshot.	Fast Snapshot Restore	\$0.75 per 1 DSU hour on each snapshot and in each Availability Zone it is enabled
Amazon EBS direct APIs for Snapshots	APIs to list and compare snapshots	ListChangedBlocks and ListSnapshotBlocks	\$0.0006 per thousand requests
Amazon EBS direct APIs for Snapshots	APIs to read snapshot blocks	GetSnapshotBlock	\$0.003 per thousand SnapshotAPIUnits
Amazon EBS direct APIs for Snapshots	APIs to write snapshot blocks	PutSnapshotBlock	\$0.006 per thousand SnapshotAPIUnits

General Purpose SSD (gp3) vs. General Purpose SSD (gp2)

AWS [announced](#) a next generation general purpose SSD at Reinvent 2020. The storage cost of gp3 is on average 20% less than gp2 (\$0.08/GB-month vs \$0.1/GB-month of GP2) for smaller storage volumes. However you must keep in mind that GP2's performance is proportional to its volume capacity, so to have higher performance, you must provision a volume with higher storage capacity. GP2 also uses the concept of "burst credit" to burst for up to 30 minutes beyond its baseline IOPS, which doesn't apply to GP3. A value of using GP3 is that you can provision its throughput and IOPS independently from its storage capacity to obtain a consistent level of desired performance, based on the following rates:

- Adding provisioned IOPS costs \$0.005/p IOPS-month
- Adding throughput costs \$0.04/ MB/s-month

So a fair comparison between the two must take into account the additional costs of GP3 performance to match the increasing performance of GP2 as the provisioned storage capacity grows. The graph below starts with a 20% differential savings that goes down as the provisioned capacity increases on the x-axis. This is due to the costs of provisioning additional GP3 performance beyond 3,000 IOPS after 1 Terabytes of capacity. At higher volumes, GP3 still remains cheaper though the difference reduces to about 5% at around 7 Terabytes of capacity.



Costs of GP2 vs GP3 while maintaining the same level of performance (savings are show on right axis)

How to lower your EBS cost

1. Find detached EBS Volumes in terminated EC2 instances

When EC2 instances are terminated, attached EBS volumes are not automatically deleted and will continue to charge since they are still operating. Check for volumes that have the AWS attribute named "state" set to the value "available", which are not attached to an EC2 instance. Also look at network throughput and IOPS to see whether there was any volume activity over the past few weeks. After you detach a volume, you are still charged for volume storage if the storage amount exceeds Amazon's Free Tier limit. You must delete a volume to avoid incurring further charges.

You can set your EC2 to automatically delete the EBS root device volume upon EC2 instance termination. Simply set *'DeleteOn Termination'* attribute for additional non-root EBS volumes to *'true'*. So, constantly check and delete any unwanted EBS volumes.

2. Find EBS volumes attached to EC2 instances stopped for too long

Amazon EC2 instances charge only while they're running, but EBS volumes attached to instances continue to retain information and charge, even when the instance is stopped. If the EC2 instances are stopped for too long (could be hours/days/months), to lower your AWS charges, it is recommended to create an EBS snapshot and delete the volume, or move your important files to an S3 bucket and delete the volume. Amazon EBS snapshots are billed at a lower rate than active EBS volumes.

3. Find underutilized EBS Volumes

Identify volumes that are underutilized (either unused storage space or IOPS), and downsize them, or change the volume type. You can monitor the read-write access of EBS volumes to determine if throughput is low. If you have a current-generation EBS volume attached to a current-generation EC2 instance type, you can use the elastic volumes feature to change the size or volume type, or (for an SSD io1 volume) adjust IOPS performance without detaching the volume.

4. Use RAID configuration to get higher IOPS

Redundant Array of Independent Disks (RAID) uses mirrored volumes to either increase performance through workload distribution or prevent single points of failure. RAID 0 or RAID 1 are recommended. RAID 0 is used to boost performance and distribute workloads when you are unable to gain additional performance by changing volume types. RAID 1 provides data redundancy.

You can reduce costs by combining multiple EBS volumes together to form a single logical volume, which is known as striping (or RAID 0). For example, assume you stripe four 120 GiB gp2 volumes together for a bursty workload in replacement of one 480 GiB io2 IOPS volume. Compare the following monthly bills:

- **4 gp3 EBS volumes:** $4 * (\$0.08 * 120 \text{ GiB}) = \38.4 ; achieves IOPS of $4 * 1920$ (16 KiB I/O)
- **4 gp2 EBS volumes:** $4 * (\$0.1 * 120 \text{ GiB}) = \48 ; achieves IOPS of $4 * 1920$ (16 KiB I/O)
- **1 io2 IOPS volume:** $\$0.125 * 480 \text{ GiB} + 3,840 * \$0.065 = \$309.6$; achieves IOPS of $2 * 1920$ (16 KiB I/O)

5. Take EBS Snapshots & Fast Snapshots Restore

EBS snapshots are typically used to create incremental backups for EBS volumes and EC2 instances. You can use EBS snapshots as part of your disaster recovery strategy and for reducing storage costs. Amazon EBS Fast Snapshot Restore enables you to create a volume from a snapshot that is fully initialized at creation. This eliminates the latency of I/O operations on a block when it is accessed for the first time. Volumes that are created using fast snapshot restore instantly deliver all their provisioned performance.

6. Remove old Snapshots that are no longer needed

Outdated Backups which have no value might still be kept in EBS snapshots. Often, recovery procedures only need the most recent snapshot for successful restoration. Although individual snapshots are not costly, the costs can add up when many are provisioned. Administrators should set up automated deletions for retained snapshots past a certain age.

7. Use Data Lifecycle Manager to manage your snapshots

You can use Amazon Data Life Cycle Manager (Amazon DLM) lifecycle policies to automate several functions for Amazon EBS snapshots. Amazon DLM eliminates the need for complicated tools and custom scripts to manage EBS snapshots. Amazon DLM enables you to manage EBS snapshots in a simple, automated way using resource tags for EBS volumes or EC2 instances. Since the operational complexity of managing EBS snapshots is reduced by Amazon DLM alone, you save money by saving your team time. Amazon DLM is free and available in all AWS Regions.

8. Moving less utilized data to lower storage tiers

Reduce the cost of less utilized data by moving to a lower grade storage like S3 or Glacier, rather than using costly EBS volumes. Since S3 pricing is a lot cheaper than the EBS or Snapshot pricing, AWS recommends using them for less utilized data.

S3 Tier	Description	Usage Category	Storage Pricing
S3 Standard	General-purpose storage for any type of data, typically frequently accessed	First 50 TB / Month	\$0.023 per GB
S3 Standard	General-purpose storage for any type of data, typically frequently accessed	Next 450 TB / Month	\$0.022 per GB
S3 Standard	General-purpose storage for any type of data, typically frequently accessed	Over 500 TB / Month	\$0.021 per GB
S3 Intelligent	Automatic cost savings for data with irregular access patterns	Frequent Access Tier, First 50 TB / Month	\$0.023 per GB
S3 Intelligent	Automatic cost savings for data with irregular access patterns	Frequent Access Tier, Next 450 TB / Month	\$0.022 per GB
S3 Intelligent	Automatic cost savings for data with irregular access patterns	Frequent Access Tier, Over 500 TB / Month	\$0.021 per GB
S3 Intelligent	Automatic cost savings for data with irregular access patterns	Infrequent Access Tier, All Storage / Month	\$0.0125 per GB
S3 Intelligent	Automatic cost savings for data with irregular access patterns	Monitoring and Automation, All Storage / Month	\$0.0025 per 1,000 objects
S3 Standard - Infrequent Access	For long-term but rarely accessed data that needs quick access	All Storage / Month	\$0.0125 per GB
S3 One Zone - Infrequent Access	For recreatable, rarely accessed data that needs quick access	All Storage / Month	\$0.01 per GB
S3 Glacier	For long-term backups and archives; retrievable within 12 hours	All Storage / Month	\$0.004 per GB
S3 Glacier Deep Archive	For long-term archives accessed once or twice per year; restorable within 12 hours	Monitoring and Automation, All Storage / Month	\$0.00099 per GB

9. Select the right mix of EBS types

AWS provides various types of EBS volumes allowing you to select the right volume to meet your budget and application performance. You should ensure that EC2 instances use General Purpose SSD (gp2) or General Purpose SSD (gp3) EBS volumes instead of Provisioned IOPS SSD (io1) volumes unless the supported application requires more than 10000 IOPS or 160 MiB/s of throughput per volume.

10. Select the right sizes of EBS volumes

EBS volumes should also be right-sized with appropriate capacity, IOPS, and throughput of the application. You should monitor the read-write access for all EBS volumes periodically. If the throughput is low, then downgrade the EBS blocks in question to reduce EBS costs.

11. Select the right size provisioned IOPS

If you're using high-performance io1 EBS volumes, capacity isn't the only dimension you'll need to optimize. In addition, the amount of provisioned IOPS should be adjusted to match application requirements. This could be a bigger cost factor than capacity. io1 is the most expensive EBS volume type because it provides the highest performance and it is configurable.

12. Select the right size RDS volumes

RDS databases often use over-provisioned EBS resources since databases are sensitive to latency. Just like volumes attached to EC2 instances, these may not be the right type or size and should be adjusted based on the performance requirements of the application.

Chapter 2: AWS Data Transfer Pricing

When using AWS for cloud computing, it's important to consider all aspects of cost. One often-overlooked cause for a growing AWS bill is the cumulative cost of data transfers. This is a particularly challenging cost to approximate due to each service having different rates and stipulations. This article will guide you through the intricacies of AWS Data Transfers and highlight some cost-effective strategies for routing your data.

What are AWS Data Transfers?

An AWS Data Transfer occurs whenever data is moved either to the Internet from AWS or moved between AWS instances across their respective Regions or Availability Zones. Generally, inbound transfers are free; inter-Region and inter-Availability Zone data transfers incur costs and are metered per Gigabyte.

Cost Scenario

In 2016 it was estimated that the average business manages over 162.9TB of data. That number was doubled for enterprise businesses. An article written in 2019 ranked AWS data transfer spending by company for the years 2017-2018. They discovered that Apple had spent close to \$50 million on data transfers in 2017. And for 7 of the top 10 biggest spenders on AWS Data transfers in 2017, costs rose by 50% in 2018.

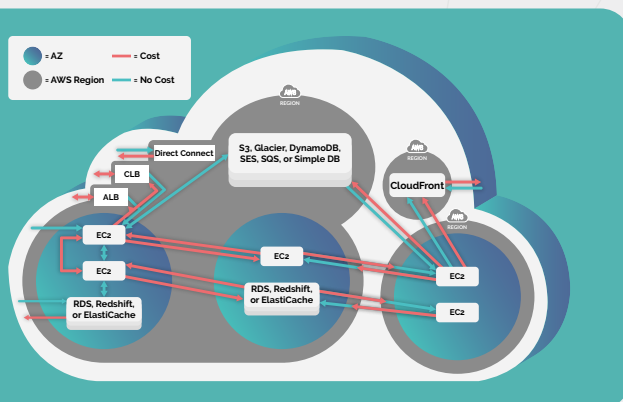
As you can see, depending on the kind of data (such as video content, or data replication) and usage patterns of your users, data transfer costs can jump significantly overnight. Planning out the most efficient flow of your data is critical to staying within budget. Fortunately, there are several ways to reduce this cost if you first understand how AWS data transfer pricing works.

AWS Data Transfer Pricing Categories

There are three common ways a business can incur data transfer costs. Let's examine each of them in detail for some of the different services available.

Data Transfer In and Out of an AWS Region

The following sections provide a quick breakdown of the cross-regional costs associated with data transfers for common Amazon services.



Direct Connect Data Transfer Pricing

AWS Direct Connect enables businesses to establish a private dedicated network connection to their AWS resources. Below is a simplified table of Direct Connect pricing from Regions within the United States to other Regions in the United States or abroad.

	USA	Canada	EU	Tokyo & Osaka	Seoul, Singapore, HK	Mumbai	Sao Paulo	Sydney	Bahrain	Cape Town
USA	\$0.02	\$0.02	\$0.03	\$0.09	\$0.09	\$0.09	\$0.15	\$0.13	\$0.11	\$0.11

EC2 Data Transfer Pricing

AWS EC2s provide resizable compute capacity in the cloud. Data transfer costs for EC2s do not vary based on whether the EC2 is a Spot, On-Demand, or Reserved instance.

Whenever you see the phrase "Standard Data Transfer rates" in AWS documentation for other services, they are referring to [On-Demand EC2 data transfer pricing](#).

Data Transfer From EC2 to Internet

OUT of EC2 to Internet	Pricing
Up to 1 GB / Month	\$0.00 per GB
Next 9.999 TB / Month	\$0.108 per GB
Next 40 TB / Month	\$0.102 per GB
Next 100 TB / Month	\$0.084 per GB
Greater than 150 TB / Month	\$0.06 per GB

Data Transfer From EC2 to Other Regions

OUT of EC2 to Internet	Pricing
Amazon Cloudfront	\$0.00 per GB
USA Regions (not Ohio)	\$0.02 per GB
USA Ohio Region	\$0.01 per GB
Non-US Regions	\$0.02 per GB

ElastiCache Data Transfer Pricing

AWS ElastiCache enables your web apps to leverage in-memory datastore (cache) in the cloud for fast information retrieval. There are no Amazon ElastiCache data transfer charges for traffic in or out of an Amazon ElastiCache node itself; charges in this case only come from traffic in or out of a given EC2 instance associated with the ElastiCache node at the standard Regional rate of \$0.01 per GB.

RDS Data Transfer Pricing

AWS RDS accelerates the setup and operation of cloud relationship databases. RDS data transfer pricing may vary per database used. Notice in the following table that there is a free pricing tier (up to 1GB/mo) for data transfers into Amazon RDS. Also, [outbound transfers to Amazon Cloudfront are completely free](#).

Data Transfer in to RDS

IN to Amazon RDS	Pricing
Up to 1 GB / Month	\$0.00 per GB
Next 9.999 TB / Month	\$0.108 per GB
Next 40 TB / Month	\$0.102 per GB
Next 100 TB / Month	\$0.084 per GB
Greater than 150 TB / Month	\$0.06 per GB

Data Transfer out of RDS

OUT of Amazon RDS	Pricing
Amazon Cloudfront	\$0.00 per GB
USA Region (not Ohio)	\$0.02 per GB
USA Ohio Region	\$0.01 per GB
Non-US Region	\$0.02 per GB

Redshift Data Transfer Pricing

Amazon Redshift is a data warehouse that enables businesses to analyze petabyte-sized data. Typically you can expect to pay standard data transfer rates when using Redshift. The only exception to this is when transferring data to or from an S3 resource within the same AWS region. If you are using a VPC, it's important to note that data transfers over JDBC/ODBC to your Amazon Redshift cluster endpoint also incur standard data transfer fees (which are the same as EC2 data transfer rates).

Data Transfers Within an AWS Region

Data transfers in or out of a given service within a specific AWS region is generally \$0.01 per GB. However, **data transfers within the same Availability Zone are free** (except for VPC Peering Connections).

Metered Data Transfers Within an AWS Region

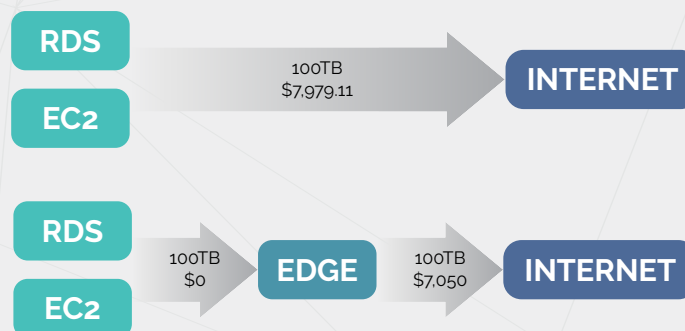
Service In Same AWS Region	Pricing
Amazon EC2	\$0.01 per GB
Amazon RDS	\$0.01 per GB
Amazon Redshift	\$0.01 per GB
Amazon DynamoDB Accelerator	\$0.01 per GB
Amazon ElastiCache	\$0.01 per GB
Elastic Network Interfaces	\$0.01 per GB
VPC Peering Connections	\$0.01 per GB

Free Data Transfers Within an AWS Region

Service In Same AWS Region	Pricing
Amazon S3	Free
Amazon Glacier	Free
Amazon DynamoDB	Free
Amazon SES	Free
Amazon SQS	Free
Amazon Kinesis	Free
Amazon ECR	Free
Amazon SNS	Free
Amazon SimpleDB	Free

Data Transfer In and Out of an AWS Edge location

AWS Edge Locations are part of the [CloudFront CDN service](#). Cloudfront enables quick delivery of media content, API actions, and other data at a preferred data transfer rate. As you can see in the following table, after the first 150TB worth of data transfers for US-based services, the cost drops by \$0.02 per GB in comparison to standard data transfer rates and continues to drop by another \$0.04 per GB.



You can achieve even better discounted data transfer rates by committing to 10TB+ of data transfers per month by contacting AWS and signing a private pricing agreement.

Per Month	USA & CA	EU & Israel	South Africa, Kenya, & Middle East	South America	Japan	Australia	Singapore, South Korea, Taiwan, Hong Kong, & Philippines	India
First 10TB	\$0.09	\$0.09	\$0.11	\$0.11	\$0.11	\$0.11	\$0.14	\$0.17
Next 40TB	\$0.08	\$0.08	\$0.11	\$0.11	\$0.09	\$0.10	\$0.14	\$0.13
Next 100TB	\$0.06	\$0.06	\$0.09	\$0.09	\$0.09	\$0.09	\$0.12	\$0.11
Next 350TB	\$0.04	\$0.04	\$0.08	\$0.08	\$0.08	\$0.09	\$0.10	\$0.10
Next 524TB	\$0.03	\$0.03	\$0.06	\$0.06	\$0.08	\$0.09	\$0.08	\$0.10
Next 4PB	\$0.03	\$0.03	\$0.05	\$0.05	\$0.07	\$0.09	\$0.07	\$0.10
Over 5PB	\$0.02	\$0.02	\$0.04	\$0.04	\$0.06	\$0.08	\$0.06	\$0.10

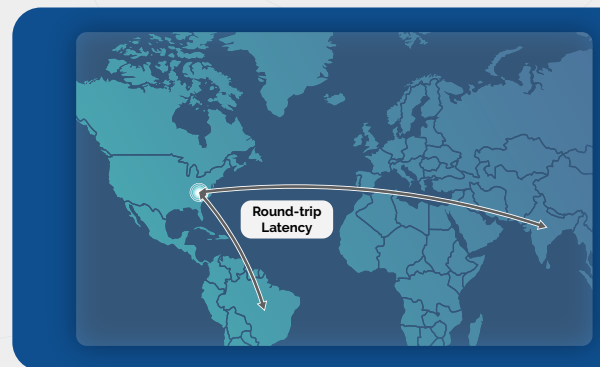
The Need For Data Transfers Explained

Some businesses may try to reduce their data transfer costs by consolidating their applications into fewer regions and availability zones. However, there are many advantages to geographic diversification.

Suppose that you are hosting a global application and have noticed a growing user base in Asia. To accelerate this growth and provide the best experience, you may need to host your application closer, perhaps in the AWS Singapore region (ap-southeast-1). This avoids long round-trip latency issues from data traveling between the US and Asia.

Another reason why you may have to use a specific region may be that you are serving the European market where corporations are required by law to store their sensitive data in data centers physically located within continental Europe. In this case, you may host an instance of your application in the region located in Frankfurt (eu-central-1) simply to comply with local regulations.

Business continuity in the event of a natural disaster or outage is also a commonly cited reason for hosting servers across different Regions or Availability Zones. This use case is further described in the next section.



Check our [glossary](#) for definitions of regions and availability zones.

How to Reduce AWS Data Transfer Costs

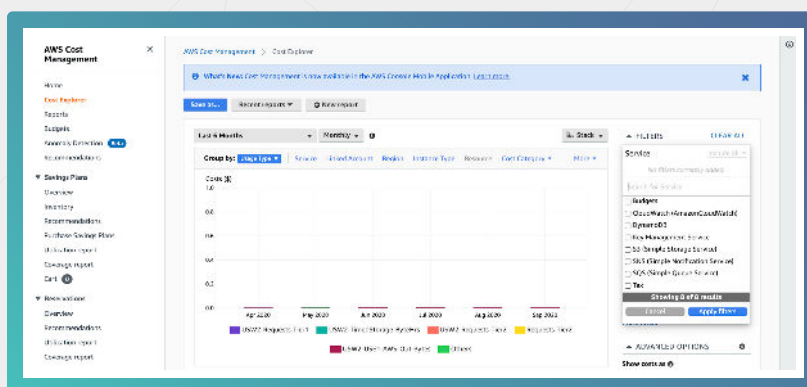
We've covered basic pricing rates for data transfers across common AWS services and why a business must seriously consider multi-regional infrastructure. Now let's look at how you can apply this information to reduce your own AWS bill.

Step 1: Understand Your Existing Data Transfer Spending

The first step towards understanding--and reducing--your AWS bill is to measure everything, discover your historical cost patterns, and visualize your spending trends. In the case of data transfer costs, you'll have to dig deep into a subset of EC2 Service line-items to get the information you need.

To find how much you are spending on data transfers:

1. Open the AWS Console.
2. to set benchmarks for month-to-month and year-to-year trend analysis. This enables you to forecast budgetary needs confidently due to the cataloging achieved in step 1.
3. available services, resources, and discount programs.
4. applies all previous steps. This plan should factor in right-sizing resources, analyzing existing commitments for utilization and efficacy, upgrading to relevant instance generations, and validating new commitments against a 1- or 3- year horizon.

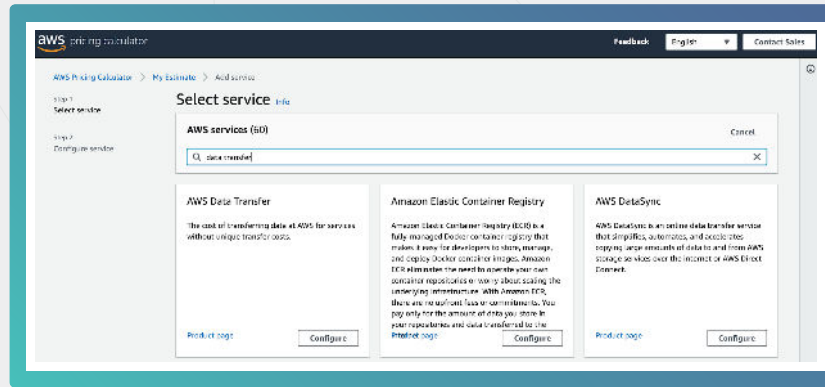


You will then be presented with a number of sub-line items that include EBS Snapshots, EBS Volume usage, and also Data Transfer costs. As an example, the data transfer line item would be labeled by a code such as "USW2-DataTransfer-Regional-Bytes".

We recommend that you take one additional step of designating the Name tag as a Cost Allocation Tag so that you can further group your spending by individual EC2 instances or Elastic Load Balancers (ELB). AWS doesn't index by default the data provided in Cost Explorer data by your AWS tags. You have to [request your tags to be designated as a Cost Allocation Tag](#). Since the "Name" of an EC2 instance or an ELB is nothing more than a tag in AWS, you can request for the Name tag to be used for indexing. This indexing request would only apply to your data going forward from the time of your request and be applied retrospectively.

Once you have isolated your data transfer costs, observe the trend over multiple months to see if you have experienced any sudden increases in recent history. You can then group by the Name tag to determine which instances contributed to the increase in your data transfer costs.

You can also price hypothetical data transfer scenarios between specific regions using the [AWS Calculator](#). Simply select **Create Estimate** and scroll to AWS Data Transfer.



Step 2: Architect Your Environment To Optimize Data Transfers

The following strategies should always be considered for your applications. Doing so ensures a proactive approach towards a healthy budget and acceptable AWS bill.

Keep Your Data Within The Same Availability Zone

We have seen cases where employees who have direct access to the AWS console launch EC2s or other AWS services in different Availability Zones (AZs) without realizing that the transfer between those nodes result in a charge. So if you don't have a reason such as ensuring disaster recovery for using different AZs, then make sure that your employees only launch new services in the same AZ. One way to ensure compliance would be to restrict their privileges with [IAM role policies](#).

Keep Your Data Within the Same Region

There are times when you must keep data in two separate AZs out of precaution, so that if one AZ goes offline, another isolated building with a separate network and power ensures your business's continuity. This said, you have the option to use two AZs that are in the same region. In terms of disaster recovery, if one AWS physical data center (AZ) is affected, then you have the protection of the second data center (or AZ) which would be located miles away. However, if you are trying to protect against a natural disaster such as an earthquake that may affect a whole region, then you would want to use two regions.

Use A Less Expensive Region

As means of context, as of the time of the writing of this article, the price of data transfer out of the Sao Paulo (Brazil) region to all other regions is \$0.14 per Gigabyte, and transfers from the Singapore region to all other regions cost \$0.09, whereas the transfer cost from the Ohio region to Atlanta is only \$0.01. So if you must use regions outside of the US, know that the international rates vary considerably. For illustration purposes, the table below shows pricing of data transfer out of a few sample regions to all other AWS regions.

OUT to all other regions	Pricing
Africa (Cape Town)	\$0.15 per GB
Brazil (Sao Paolo)	\$0.14 per GB
Middle East (Bahrain)	\$0.11 per GB
Asia (Singapore)	\$0.09 per GB
Asia (Seoul)	\$0.08 per GB
GovCloud (East)	\$0.03 per GB
Europe (Frankfurt)	\$0.02 per GB
UK (London)	\$0.02 per GB
US (Ohio) to US (N. VA)	\$0.01 per GB

Optimize Your Use of Public IP Addresses

There are rules that apply to data transfer pricing when using a public IP address or an Elastic IPv4 or IPv6. The rules are best explained in the Data Transfer section of the [AWS page dedicated to explaining EC2 pricing](#) in small fonts. Data transferred "in" to and "out" from a public or Elastic IPv4 address is charged at \$0.01/GB in each direction, while data transferred "in" to and "out" from an IPv6 address in a different VPC is charged at \$0.01/GB in each direction. You can optimize your spending by using a Private IP address when transferring data that doesn't leave your region or AZ.

Use AWS CloudFront

Using a content delivery network (CDN) such as CloudFront is essential in keeping data transfer costs low. CloudFront moves your most accessed assets to the "front" of the AWS network (i.e., [Edge Locations](#)), ensuring fast delivery to your end users. Data transfers into AWS CloudFront from other AWS resources are generally free; Data transfers from AWS CloudFront to the internet are **20-40% cheaper than standard data transfer rates**--with the possibility of deeper discounts through a Private Pricing Program contract.

Step 3: Sign up for the AWS Data Transfer Private Pricing Program

You can reduce your data transfer costs by committing to an annual volume in advance. This volume strategy is conceptually similar to Reserved Instances (RI) or Savings Plans, and it's referred to as a Private Pricing Program.

For instance, let's say that you have concluded that you need to consume 1 Petabyte's worth of data transfer each month over the next 12 months. You are unable to reduce that cost any further by optimizing or re-architecting. In that case, you would enter into a Private Pricing Program and commit to using a minimum of 1 Petabyte of data transfer monthly across designated regions between AWS services such as EC2, RDS, ElastiCache, or Redshift.

If you do not use the data transfer amount that you have committed to, you are still obligated to pay for the full committed amount.

In exchange for the commitment, you will receive a significant discount which is negotiated between you and your AWS account manager and which must remain confidential. Since the private pricing agreements are negotiated and signed on a case by case basis, you won't find any standard volume discount pricing schedules associated with the program. We recommend that you reach out to your AWS account manager to start the process.

Chapter 3: AWS Disaster Recovery Pricing

In this article we review various disaster recovery strategies available in Amazon Web Services (AWS) and we explain the pricing associated with each option.

What is AWS Disaster Recovery?

Preventing unforeseeable disasters that cause business outages and product loss is one of the most significant challenges for IT experts today. The AWS Disaster Recovery Plan (DRP) helps you store and restore data to minimize disasters that may cause loss of infrastructure, plans, and data. While there are many DR strategy initiatives in the market, AWS offers several service options within its own ecosystem that ensure business continuity.

Why do we use AWS Disaster Recovery?

- Financial low cost
- High performance
- High elasticity
- Effective handling
- Fast setup time
- High availability
- Fewer dependencies
- Easy testing
- Flexible locations
- All on a single cloud
- Industry standards
- Effective monitoring
- Scalability for services
- Ready and standby
- Efficient backup
- Secure user access
- Security
- High reliability
- Easy recovery
- Automation

Top 10 Disaster Recovery Use Cases & AWS Solutions

1. **DR for applications hosted in AWS:** [AWS Regions](#), [Availability Zones](#)
2. **DR for applications hosted outside of AWS:** [DataSync](#), [AWS Import/Export](#)
3. **Data Backup and Restore:** [AWS BackUp](#), [AWS Import/Export](#), [EBS](#)
4. **Business Continuity Planning (BCP):** [Amazon WorkSpaces](#), [AWS BackUp](#), [CloudEndure](#), [S3](#)
5. **Data Lakes and Analytics:** [Data Movement](#), [Data Lake](#), [Analytics](#), [Machine Learning](#)
6. **Infrastructure Modernization:** [Server Migration Service](#), [RDS](#), [Elastic Beanstalk](#)
7. **Data Archive:** [S3 Storage Classes](#), [Storage Gateway](#), [Snow Family](#), [DataSync](#), [Glacier](#)
8. **Data Migration and Transfer:** [Application Discovery Service](#), [DataSync](#), [Database Migration Service](#), [Server Migration Service](#)
9. **Data Replication:** [RDS](#), [DynamoDB](#), [S3](#), [EC2](#), [EC2 VM Import Connector](#), [CloudFormation](#)
10. **Data Protection:** [Direct Connect](#), [VPC](#), [Elastic Load Balancing](#), [Route 53](#)

How to Measure Disaster Recovery

There are two main measurements to consider when developing a disaster recovery plan:

1. **Recovery Point Objective (RPO):** defines the acceptable maximum time interval (representing potential data loss) between creating recovery points.
2. **Recovery Time Objective (RTO):** defines the acceptable maximum delay between the interruption and restoration of service.

AWS Disaster Recovery Methods

1. **Backup and Restore:** This method is a simple and straightforward way to backup and restore data as needed, however, it can be time-consuming due to none of your data being on standby.
2. **Pilot Light:** This method keeps critical applications and data at-the-ready so that it can be quickly retrieved if needed.
3. **Warm Standby:** This method always keeps a duplicate version of your business' core elements running on standby, which makes for a little downtime and an almost seamless transition.
4. **Multi-Site Solution:** This method fully replicates the data/applications between two or more active locations and splits your traffic between them. Interruptions simply trigger traffic to be rerouted to the unaffected area, resulting in almost zero downtime. But running two separate environments incurs much higher costs.

Recovery Method Comparison

Metric	Backup & Restore	Pilot Light	Warm Standby	Multi-Site
RTO/RPO	Hours	10s of Minutes	Minutes	Real-Time
Usage	Lower priority use cases	Lower RTO/RPO requirements	Core applications and services	Mission-critical applications and services
Solutions	Cloud storage, Backup solutions	Database services, Replication solutions	Cloud storage, Databases services, Replication solutions	Database services, Replication solutions
Cost	\$ to \$\$	\$\$	\$\$\$	\$\$\$\$

Disaster Recovery Solutions Offered by AWS

In this section, we will review the services offered by AWS intended for disaster recovery planning. We also explain pricing associated with each offering and review pricing examples in cases where the pricing is more complex.

The AWS services covered in this section are:

1. RDS Replication (Aurora, PostgreSQL, MySQL, Maria, Oracle)
2. Non-relational Database Replication (DynamoDB)
3. S3 object storage replication (Cross-region, same-region, replication time control)
4. File system replication (DataSync)
5. Backup
6. Storage Gateway
7. Snow family
8. CloudEndure (Block storage replication)
9. Transfer family
10. Database and server migration service

Amazon Relational Database (RDS) Replication

Amazon RDS makes it easy to manage relational databases in the cloud. It provides cost-efficient, resizable capacity and automates administrative tasks like hardware provisioning, database setup, patching, and backups. Amazon RDS is available on several database instance types optimized for memory, performance, or I/O and provides you with six database engines to choose from, including Amazon Aurora, PostgreSQL, MySQL, MariaDB, Oracle Database and SQL Server.

Aurora Data Replication

AWS provides three replication options with Aurora:

1. **Aurora replicas:** Aurora Replicas act as independent endpoints in an Aurora DB cluster and are primarily used to scale read operations or increase availability. Up to 15 Aurora Replicas can be used across any Availability Zones covered by a DB cluster within an AWS Region.
2. **Aurora MySQL replication:** There are 3 replication options for Aurora MySQL.
 3. Two DB clusters in different AWS Regions.
 4. Two DB clusters in the same AWS Region.
 5. One master Amazon RDS MySQL DB instance with an Aurora MySQL DB cluster.
3. **Aurora PostgreSQL replication:** Set up replication between a master Amazon RDS PostgreSQL DB instance with an Aurora PostgreSQL DB cluster.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
Database Instances – MySQL Compatible	On-Demand Instance	\$0.041 per Hour	\$0.048 per Hour
Database Instances – MySQL Compatible	T3 CPU Credits	\$0.09 per vCPU Hour	\$0.09 per vCPU Hour
Database Instances – MySQL Compatible	Serverless	\$0.06 per Aurora Capacity Unit Hour	\$0.07 per Aurora Capacity Unit Hour
Database Instances – MySQL Compatible	Reserved Instance	\$0.033 per Hour	\$0.036 per Hour
Database Instances – PostgreSQL Compatible	On-Demand Instance	\$0.082 per Hour	\$0.096 per Hour
Database Instances – PostgreSQL Compatible	Serverless	\$0.06 per Aurora Capacity Unit Hour	\$0.07 per Aurora Capacity Unit Hour
Database Instances – PostgreSQL Compatibl	Reserved Instance	\$0.064 per Hour	\$0.072 per Hour
Database Storage and IOs	Storage Rate	\$0.10 per GB-month	\$0.119 per GB-month
Database Storage and IOs	I/O Rate	\$0.20 per 1 million requests	\$0.22 per 1 million requests
Global Database – Replicated Write I/Os	Replicated Write I/Os	\$0.20 per million replicated write I/Os	\$0.22 per million replicated write I/Os
Backup Storage	Backup Storage	\$0.021 per GB-month	\$0.023 per GB-month
Backtrack	Change Records	\$0.012 per 1 million Change Records	\$0.014 per 1 million Change Records
Snapshot Export	Charge per GB of snapshot size:	\$0.01 per GB	\$0.011 per GB
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB Month	\$0.00 per GB Month
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB	\$0.02 per GB

PostgreSQL Data Replication

AWS provides three replication options with PostgreSQL:

1. **Intra-Region replication:** RDS PostgreSQL uses Postgres native streaming replication to create a read-replica for source instances in the same AWS Region. Any data changes are streamed to the read-replica using the streaming replication.
2. **Cross-region replication:** RDS PostgreSQL also supports cross-region replication. In addition to scaling read queries, cross-region read-replicas provide solutions for disaster recovery and database migration between AWS Regions.
3. **Logical replication:** You can set up logical replication slots at an RDS PostgreSQL instance and stream database changes. AWS Database Migration Service (AWS DMS) exhibits the most common use case of logical replication.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
On-Demand DB Instances	Single-AZ Deployment	\$0.018 per Hour	\$0.021 per Hour
On-Demand DB Instances	Multi-AZ Deployment	\$0.036 per Hour	\$0.042 per Hour
On-Demand DB Instances	T3 CPU Credits	\$0.075 per vCPU Hour	\$0.075 per vCPU Hour
Reserved Instances	Single-AZ Deployment	\$0.013 per Hour	\$0.014 per Hour
Reserved Instances	Multi-AZ Deployment	\$0.026 per Hour	\$0.029 per Hour
General Purpose (SSD) Storage in Single AZ	Storage Rate	\$0.125 per GB-month	\$0.149 per GB-month
General Purpose (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.10 per IOPS-month	\$0.119 per IOPS-month
General Purpose (SSD) Storage in Single AZ	Storage Rate	\$0.20 per GB-month	\$0.238 per GB-month
General Purpose (SSD) Storage in Single AZ	I/O Rate	\$0.10 per 1 million requests	\$0.11 per 1 million requests
Backup Storage	Backup Storage	\$0.103 per GB-month	\$0.103 per GB-month
Snapshot Export	Charge per GB of snapshot	\$0.0100	\$0.0110
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB per Month	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB per Month	\$0.02 per GB per Month

MySQL Data Replication

AWS provides three replication options with MySQL:

1. **MySQL Read Replicas:** You can create up to 5 read-replicas from one DB instance. For replication to operate effectively, each read-replica should have the same amount of compute and storage resources as the source DB instance.
2. **GTID-based Replication:** You can use GTID-based replication to replicate data with Amazon RDS MySQL read-replicas or with an external MySQL database.
3. **MySQL or MariaDB Replication:** You can set up replication between an Amazon RDS MySQL or MariaDB DB instance and a MySQL or MariaDB instance that is external to Amazon RDS.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
On-Demand DB Instances	Single-AZ Deployment	\$0.017 per Hour	\$0.02 per Hour
On-Demand DB Instances	Multi-AZ Deployment	\$0.034 per Hour	\$0.04 per Hour
On-Demand DB Instances	T3 CPU Credits	\$0.075 per vCPU-Hour	\$0.075 per vCPU-Hour.
Reserved Instances	Single-AZ Deployment	\$0.012 per Hour	\$0.014 per Hour
Reserved Instances	Multi-AZ Deployment	\$0.024 per Hour	\$0.029 per Hour
General Purpose (SSD) Storage	Single-AZ Deployment	\$0.115 per GB-month	\$0.137 per GB-month
General Purpose (SSD) Storage	Multi-AZ Deployment	\$0.23 per GB-month	\$0.273 per GB-month
Provisioned IOPS (SSD) Storage in Single AZ	Storage Rate	\$0.125 per GB-month	\$0.149 per GB-month
Provisioned IOPS (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.10 per IOPS-month	\$0.119 per IOPS-month
Provisioned IOPS (SSD) Storage in Single AZ	Storage Rate	\$0.25 per GB-month	\$0.298 per GB-month

Provisioned IOPS (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.20 per IOPS-month	\$0.238 per IOPS-month
Magnetic Storage in Single AZ	Storage Rate	\$0.10 per GB-month	\$0.119 per GB-month
Magnetic Storage in Single AZ	I/O Rate	\$0.10 per 1 million requests	\$0.11 per 1 million requests
Magnetic Storage in Single AZ	Storage Rate	\$0.20 per GB-month	\$0.238 per GB-month
Magnetic Storage in Single AZ	I/O Rate	\$0.10 per 1 million requests	\$0.11 per 1 million requests
Backup Storage	Backup Storage	\$0.103 per GB-month	\$0.103 per GB-month
Snapshot Export	Charge per GB of snapshot size:	\$0.0100	\$0.0110
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB	\$0.02 per GB

MariaDB Data Replication

AWS provides four replication options with MariaDB:

1. **MariaDB Read-Replicas:** You can create up to 5 read-replicas from one DB instance. For replication to operate effectively, each read-replica should have the same amount of compute and storage resources as the source DB instance.
2. **GTID-based Replication:** You can set up GTID-based replication from an external MariaDB instance of version 10.0.24 or greater into an Amazon RDS MariaDB DB instance.
3. **Replication with MySQL/MariaDB instances running external to Amazon RDS:** You can set up replication between an Amazon RDS MariaDB DB instance and a MySQL or MariaDB instance that is external to Amazon RDS.
4. **Importing to an Amazon RDS MySQL/MariaDB DB instance with reduced downtime and exporting data from a MySQL DB:** You can configure replication to import databases from a MySQL or MariaDB instance that is external to Amazon RDS, or to export databases to such instances.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
On-Demand DB Instances	Single-AZ Deployment	\$0.017 per Hour	\$0.02 per Hour
On-Demand DB Instances	Multi-AZ Deployment	\$0.034 per Hour	\$0.04 per Hour
On-Demand DB Instances	T3 CPU Credits	\$0.075 per vCPU-Hour	\$0.075 per vCPU-Hour
Reserved Instances	Single-AZ Deployment	\$0.012 per Hour	\$0.014 per Hour
Reserved Instances	Multi-AZ Deployment	\$0.024 pe Hour	\$0.029 per Hour
General Purpose (SSD) Storage	Single-AZ Deployment	\$0.115 per GB-month	\$0.137 per GB-month
General Purpose (SSD) Storage	Multi-AZ Deployment	\$0.23 per GB-month	\$0.273 per GB-month
Provisioned IOPS (SSD) Storage in Single AZ	Storage Rate	\$0.125 per GB-month	\$0.149 per GB-month
Provisioned IOPS (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.10 per IOPS-month	\$0.119 per IOPS-month
Provisioned IOPS (SSD) Storage in Single AZ	Storage Rate	\$0.25 per GB-month	\$0.298 per GB-month
Provisioned IOPS (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.20 per IOPS-month	\$0.238 per IOPS-month
Magnetic Storage in Single AZ	Storage Rate	\$0.10 per GB-month	\$0.119 per GB-month

Magnetic Storage in Single AZ	I/O Rate	\$0.10 per 1 million requests	\$0.11 per 1 million requests
Magnetic Storage in Single AZ	Storage Rate	\$0.20 per GB-month	\$0.238 per GB-month
Magnetic Storage in Single AZ	I/O Rate	\$0.10 per 1 million requests	\$0.11 per 1 million requests
Backup Storage	Backup Storage	\$0.103 per GB-month	\$0.103 per GB-month
Snapshot Export	Charge per GB of snapshot size:	\$0.0100	\$0.0100
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB	\$0.02 per GB

Oracle Data Replication

AWS provides two replication options with Oracle:

1. **Read-only Replicas:** This is the default and Active Data Guard transmits and applies changes from the source database to all read replica databases. You can create up to 5 read-replicas from one source DB instance.
2. **Mounted Replicas:** The replication uses Oracle Data Guard, but the replica database doesn't accept user connections. The primary use for mounted replicas is cross-region disaster recovery.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
On-Demand DB Instances (License Included)	Single AZ	\$0.035 per Hour	\$0.038 per Hour
On-Demand DB Instances (License Included)	Multi AZ	\$0.07 per Hour	\$0.076 per Hour
On-Demand DB Instances (License Included)	T3 CPU Credits	\$0.075 per vCPU-Hour	\$0.075 per vCPU-Hour
On-Demand DB Instances (Bring Your Own License)	Single AZ	\$0.017 per Hour	\$0.02 per Hour
On-Demand DB Instances (Bring Your Own License)	Multi AZ	\$0.034 per Hour	\$0.04 per Hour
On-Demand DB Instances (Bring Your Own License)	T3 CPU Credits	\$0.075 per vCPU-Hour	\$0.075 per vCPU-Hour
Reserved Instances (License Included)	Single AZ	\$0.022 per Hour	\$0.027 per Hour
Reserved Instances (License Included)	Multi AZ	\$0.044 per Hour	\$0.055 per Hour
Reserved Instances (Bring Your Own License)	Single AZ	\$0.011 per Hour	\$0.015 per Hour
Reserved Instances (Bring Your Own License)	Multi AZ	\$0.021 per Hour	\$0.029 per Hour
General Purpose (SSD) Storage in Single AZ	General Purpose (SSD) Storage	\$0.115 per GB-month	\$0.137 per GB-month
General Purpose (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.125 per GB-month	\$0.149 per GB-month
General Purpose (SSD) Storage in Single AZ	Magnetic Storage	\$0.10 per GB-month	\$0.119 per GB-month
General Purpose (SSD) Storage in Multi AZ	General Purpose (SSD) Storage	\$0.23 per GB-month	\$0.273 per GB-month
General Purpose (SSD) Storage in Multi AZ	Provisioned IOPS Rate	\$0.25 per GB-month	\$0.298 per GB-month
General Purpose (SSD) Storage in Multi AZ	Magnetic Storage	\$0.20 per GB-month	\$0.238 per GB-month
Backup Storage	Backup Storage	\$0.103 per GB-month.	\$0.103 per GB-month
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB	\$0.02 per GB

SQL Server Data Replication

AWS provides one replication option with SQL Server:

1. **MySQL Read-Replicas:** Use these read-replicas to configure replication between Amazon RDS DB instances.

DB Service	Pricing Option	US East (N. Virginia) Pricing	Europe (Frankfurt) Pricing
On-Demand DB Instances	Express	\$0.044 per Hour	\$0.05 per Hour
On-Demand DB Instances	Web	\$ 0.139 per Hour	\$0.143 per Hour
On-Demand DB Instances	Standard	\$1.044 per Hour	\$1.081 per Hour
On-Demand DB Instances	Enterprise	\$2.262 per Hour	\$2.295 per Hour
Reserved Instances	Express	\$0.028 per Hour	\$0.032 per Hour
Reserved Instances	Web	\$0.088 per Hour	\$0.09 per Hour
Reserved Instances	Standard	\$0.987 per Hour	\$1.022 per Hour
Reserved Instances	Enterprise	\$2.242 per Hour	\$2.291 per Hour
General Purpose (SSD) Storage in Single AZ	General Purpose (SSD) Storage	\$0.115 per GB-month	\$0.137 per GB-month
General Purpose (SSD) Storage in Single AZ	Provisioned IOPS Rate	\$0.125 per GB-month	\$0.149 per GB-month
General Purpose (SSD) Storage in Single AZ	Magnetic Storage	\$0.10 per GB-month	\$0.119 per GB-month
General Purpose (SSD) Storage in Multi AZ	General Purpose (SSD) Storage	\$0.23 per GB-month	\$0.273 per GB-month
General Purpose (SSD) Storage in Multi AZ	Provisioned IOPS Rate	\$0.25 per GB-month	\$0.298 per GB-month
General Purpose (SSD) Storage in Multi AZ	Magnetic Storage	\$0.20 per GB-month	\$0.238 per GB-month
Backup Storage	Backup Storage	\$0.103 per GB-month.	\$0.103 per GB-month
Data Transfer	Data Transfer IN to Amazon RDS from Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to Internet	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data Transfer OUT from Amazon RDS to ...	\$0.02 per GB	\$0.02 per GB

Amazon Non-Relational Database Replication

Amazon DynamoDB is a non-relational database for storing key-value pairs and documents that delivers single-digit millisecond performance at any scale. It's a fully managed, multi-regional, multi-meter, durable database with built-in security, backup, restore, and in-memory caching for internet-scale applications.

AWS DynamoDB Data Replication

AWS provides two replication options with DynamoDB:

1. **Cross-Region Replication:** Create tables that get automatically replicated across other AWS Regions, with full support for multi-active writes. This service enables propping up massive applications for a global user base without having to manage the replication process.
2. **Multi-Region Replication:** Amazon DynamoDB Global Tables provide a fully managed solution for deploying a multi-regional, multi-active database, without having to build and maintain your own replication solution.

DB Service	Pricing Option	On-Demand Capacity – US East (N. Virginia) Pricing	On-Demand Capacity – Europe (Frankfurt) Pricing	Provisioned Capacity – US East (N. Virginia) Pricing	Provisioned Capacity – Europe (Frankfurt) Pricing
Read and write requests	Write request units	\$1.25 per million write request units	\$1.525 per million write request units	\$0.00065 per WCU	\$0.000793 per WCU
Read and write requests	Read request units	\$0.25 per million read request units	\$0.305 per million read request units	\$0.00013 per RCU	\$0.0001586 per RCU
Data Storage	Raw byte size of the data you upload plus a per-item storage overhead of 100 bytes to account for indexing	First 25 GB stored per month is free. \$0.25 per GB-month thereafter.	First 25 GB stored per month is free. \$0.306 per GB-month thereafter.	First 25 GB stored per month is free. \$0.25 per GB-month thereafter.	First 25 GB stored per month is free. \$0.306 per GB-month thereafter.
Backup and restore	Continuous backups (PITR)	\$0.20 per GB-month	\$0.2448 per GB-month	\$0.20 per GB-month	\$0.2448 per GB-month
Backup and restore	On-demand backup	\$0.10 per GB-month	\$0.1224 per GB-month	\$0.10 per GB-month	\$0.1224 per GB-month
Backup and restore	Restoring a table	\$0.15 per GB	\$0.1836 per GB	\$0.15 per GB	\$0.1836 per GB
Global tables	Replicated write request unit	\$1.875 per million replicated write request units	\$2.2875 per million replicated write request units	\$0.000975 per rWCU per hour	\$0.0011895 per rWCU per hour
DynamoDB Accelerator (DAX)	DAX capacity	\$0.04 Per Hour	\$0.045 Per Hour	\$0.04 Per Hour	\$0.045 Per Hour
DynamoDB Streams	Read request units	The first 2,500,000 DynamoDB Streams read request units are free. \$0.02 per 100,000 thereafter.	The first 2,500,000 DynamoDB Streams read request units are free. \$0.0245 per 100,000 thereafter.	The first 2,500,000 DynamoDB Streams read request units are free. \$0.02 per 100,000 thereafter.	The first 2,500,000 DynamoDB Streams read request units are free. \$0.0245 per 100,000 thereafter.
Data Transfer	Data transfer IN	\$0.00 per GB	\$0.00 per GB	\$0.00 per GB	\$0.00 per GB
Data Transfer	Data transfer OUT	\$0.09 per GB	\$0.09 per GB	\$0.09 per GB	\$0.09 per GB

Amazon Object Storage Replication

Amazon Simple Storage Service (S3) Replication is a low-cost feature that replicates objects between buckets while offering the most flexibility and functionality for cloud storage. You can even configure Amazon S3 to automatically replicate S3 objects across different AWS Regions.

- Amazon S3 Cross-Region Replication (CRR):** You can replicate objects, including all of their metadata and tags, into other AWS Regions for disaster prevention. S3 CRR replicates objects from a source bucket in one AWS Region into a destination bucket located in another AWS Region.
- Amazon S3 Same-Region Replication (SRR):** SRR automatically replicates data between buckets from the same AWS Region. You can set up replication at several levels using S3 object tags. SRR also supports making a second copy of your data within the same AWS Region.
- Amazon Replication Time Control:** Amazon S3 replication time control helps business requirements for data replication and provides visibility into Amazon S3 replication activity. Replication time control replicates most objects that you upload to Amazon S3 in seconds, and 99.99% of those objects within 15 minutes.

Simple Storage Replication Pricing

For S3 Cross-Region Replication and Same Region Replication, you pay the S3 charges for storage in the selected destination S3 storage class, the storage charges for the primary copy, replication PUT requests and applicable infrequent access storage retrieval fees. For CRR, you also pay for inter-region Data Transfer OUT from S3 to your destination region. When you use S3 Replication Time Control, you also pay a Replication Time Control Data Transfer fee (\$0.015 per GB) and S3 Replication Metrics charges that are billed at the same rate as Amazon CloudWatch custom metrics. Storage and PUT request pricing for the replicated copy is based on the destination AWS Region, while pricing for inter-region data transfers is based on the source AWS Region. Refer to [AWS S3 pricing](#) for more information.

Example:

- **Source S3 bucket (N. Virginia):** 500 GB
- **Destination Region:** US West (N. California)
- **PUT requests at destination:** 500
- **S3 Standard storage cost:** 500 GB * \$0.023 = \$11.50
- **S3 Standard storage cost for replicated data:** 500 GB * \$0.023 = \$11.50

- **Data transfer costs:** $500 \text{ GB} * \$0.02 \text{ (per GB data transferred)} = \10.00
- **PUT request pricing:** $\$0.005 \text{ (per 1000 requests)} / 1000 = \0.000005
- **Cost of replicated PUT requests:** $500 * \$0.000005 = \0.0025
- **Total:** $\$11.50 + \$11.50 + \$10.00 + \$0.0025 = \$33.025$

AWS File System Replication

[AWS DataSync](#) enables quick migration for large amounts of data online between on-premises storage and Amazon S3, Amazon Elastic File System (EFS) or Amazon FSx for Windows File Server. DataSync eliminates or automatically handles many tasks, such as: scripting copy jobs, scheduling and monitoring transfers, validating data, and optimizing network utilization.

DataSync Pricing

Example 1: Archive 100 TB into Amazon S3

If you migrate 100 TB of files into Amazon S3 (US East Ohio), with an average individual file size of 16 MB, it would cost you the following to use DataSync:

- $(100 \text{ TB copied into S3} * 1024 \text{ GB} * \$0.0125/\text{GB}) + (1 \text{ S3 LIST request} * \$0.005 / 1000) + (100 \text{ TB} / 16 \text{ MB S3 PUT requests} * \$0.005 / 1000)$
- = $\$1280 + \$0 + \$16.38$
- = $\$1312.76$

Example 2: Migrate 100 TB to Amazon FSx for Windows File Server

If you migrate 100 TB from your Windows File Servers hosted in your own data center to Amazon FSx, it would cost you the following to use DataSync:

- $(100 \text{ TB copied into Amazon FSx for Windows File Server} * 1024 \text{ GB} * \$0.0125/\text{GB})$
- = $\$1,280.00$

Example 3: Ongoing data transfers once your data is hosted in AWS

If you are hosted in AWS and transfer 1 TB of data into Amazon EFS, your cost for a month with 31 days would be:

- $(1 \text{ TB copied into EFS} * 1024 \text{ GB} * \$0.0125/\text{GB}) * 31 \text{ days}$
- = $\$396.80$

Example 4: If you replicate your files for the purpose of disaster recovery

If you setup a data replication job that copies a 100 TB file system into Amazon EFS in Asia Pacific (Sydney) once and then replicates 1 TB of changes every day, your charges for the initial data transfer and the 31 following days would be:

- $100 \text{ TB} * 1024 \text{ GB} * \$0.0125/\text{GB} = \$1280$ for the first bulk data transfer
- $+ 1 \text{ TB} * 1024 \text{ GB} * \$0.0125/\text{GB} * 31 \text{ days}$
- = $\$396.80$ per month for each following months

AWS Backup

[AWS Backup](#) is a fully managed backup service that centralizes and automates the backup of data across AWS services. Use AWS Backup to configure backup policies and monitor backup activity for resources such as Amazon EBS volumes, Amazon EC2 instances, Amazon RDS databases, Amazon DynamoDB tables, Amazon EFS file systems and AWS Storage Gateway volumes.

AWS Backup Pricing

- **Storage Pricing:** Billing is based on the average storage space used in a given month (in GB).
- **Restoration Pricing:** Billing is based on the amount of data restored in a given month (in GB).

Example:

Let's assume that you have instances of Amazon EFS in the US East (N. Virginia) region. If you back up 400 GB of Amazon EFS storage during the first 15 days of the month, and then another 1 TB for the last 15 days of the same month, you would have the following backup storage costs:

- **Total usage (GB-Hours):** $[500 \text{ GB} \times 15 \text{ days} \times (24 \text{ hours} / \text{day})] + [1000 \text{ GB} \times 15 \text{ days} \times (24 \text{ hours} / \text{day})] = 540,000$ GB-Hours

We then convert GB-Hours to GB-Months by dividing it by 720, which is the average number of hours in a month to obtain the value of 750 GB-Months:

- **Total Monthly Storage Charge:** $750 \text{ GB-Month} \times \$0.05 = \$37.50$

Now, let's assume you restored 10 backups of 1GB during that same month. At the end of the month, you would have the following backup restore volume, in GB:

- **Total Usage:** $10 \text{ restores} \times 1 \text{ GB} = 10 \text{ GB}$
- **Total Monthly Charges:** $10 \text{ GB} \times \$0.02 = \0.20
- **Total Monthly AWS Backup Bill:** $\$37.50 + \$0.20 = \$37.70$

AWS Storage Gateway

[AWS Storage Gateway](#) is a hybrid-cloud storage service that provides on-premises access to near-unlimited cloud storage. You can use Storage Gateway to simplify storage management and reduce costs for several use cases such as: migrating backups to the cloud, using local file shares backed by cloud storage, and providing low-latency access to data for local applications. [AWS Storage Gateway pricing](#) listed here.

AWS Snow Family

[AWS Snow Family](#) helps run operations in modest, non-data center environments and in locations with inconsistent network connectivity. These services assist in physically transporting data in and out of AWS and consists of AWS Snowcone, AWS Snowball, and AWS Snowmobile. Snow Family devices are owned by AWS and compliment AWS security, monitoring, storage management, and computing capabilities.

Snow Family Pricing:

Refer to individual pricing and examples of [AWS Snowcone](#), [AWS Snowball](#), and [AWS Snowmobile](#).

CloudEndure Disaster Recovery

[CloudEndure Disaster Recovery](#) continuously replicates machines, including their operating system, their system state configuration(s), any databases or applications, and all files into a staging area in your target AWS account and Region. With CloudEndure Disaster Recovery, you can automatically launch thousands of your machines in their fully provisioned state in minutes. CloudEndure Disaster Recovery is often used to protect critical databases and enterprise applications.

CloudEndure Pricing:

Hourly rate per source server is \$0.028 per-host, per-hour.

AWS Transfer Family

[AWS Transfer Family](#) supports file transfers directly in and out of Amazon S3. This service supports Secure File Transfer Protocol (SFTP), File Transfer Protocol over SSL (FTPS), and File Transfer Protocol (FTP).

AWS Database and Server Migration Service

[AWS Database Migration Service](#) supports database migration into AWS while keeping the source database fully operational during the process. This minimizes downtime to reliant applications.

This service is agentless, making it easier to migrate on-premise workloads into AWS. You can also automate, schedule, and track the replications of live server volumes.

Disaster Recovery and AWS Reservation

A common question that arises in the context of AWS disaster recovery is whether Reserved Instances (RI) help with planning. The answer is that only a [zonal reservation](#) defined as a reservation made with a scope limited to a specific Availability Zone (AZ), sets aside capacity inside of the AWS data center. In other words, if a disaster strikes a specific AWS Availability Zone (defined as a specific physical data center location) which in turn would increase demand for capacity in other AWS regions and Availability Zones, then only a zonal reservation would guarantee room for your workload in a new AZ. Any other types of reservation or [Savings Plan](#) contracts will help you lower your spending, but are not designed to reserve capacity in an AWS data center to help with disaster recovery planning.

Conclusion

As you can see, there are many ways to protect your service quality and data in AWS. The most common option is to replicate your data so that you avoid any loss during an outage. The right solution for your use case depends on your budget as well as your application's mission criticality and architecture.

Chapter 4: AWS Savings Plan

Over the past decade, AWS has evolved its savings opportunities to fit most common business cases across various size requirements—everything from bargain ephemeral compute capacity (Spot) to efficient long-term storage (Glacier). The most well-known of these opportunities has been commitment discounts through reserving instances (RIs). This article will look at how [AWS Savings Plans](#) differ from RIs and why you should use both to create a balanced savings portfolio.

Before we dive into AWS Savings Plans, let's quickly review [AWS Reserved Instances](#). Over the years, RIs have saved businesses millions of dollars through the following discounts:

1. **Regional Benefits:** Applies RIs across all Availability Zones in a region.
2. **Convertible RIs:** Allows for chan
3. **Instance Size Flexibility:** Allows your Regional RIs to apply to any instance size within an instance family.

The RI discount model can provide discounts of up to 72%, but it does require you to coordinate your RI purchases and exchanges to ensure that you have an optimal mix that covers usage, which might change over time.

What is an AWS Savings Plan?

AWS Savings Plans offer significant savings over [On-Demand EC2s](#) in exchange for a commitment to use a specific amount of computing power. This rate is measured by dollars-per-hour over a 1-year or 3-year period. You can sign up for AWS Savings Plans through [AWS Cost Explorer](#).

AWS Savings Plans can provide savings of up to 72% on your AWS compute usage—regardless of instance family, size, OS type, tenancy, or AWS Region. These discounts can also be applied to [AWS Fargate](#) and [AWS Lambda](#) usage as well.

It's important to note that any additional usage on top of your initial commitment is charged at the standard On-Demand rate. AWS Savings Plans are not applied to any usage of existing Reserved Instances.

So far, AWS Savings Plans may sound quite similar to Reserved Instances—so what's the difference? The answer: using AWS Savings Plans provides some much-needed flexibility. We'll break down exactly what that flexibility looks like in the following sections.

Types of AWS Savings Plans

AWS offers two types of Savings Plans.

1. Compute Savings Plans

Compute Savings Plans provide the most flexibility and help to reduce your costs by up to 66% (similar to Convertible RIs). These plans automatically apply to any EC2 instance usage regardless of region, instance family, operating system, or tenancy—including those that are part of EMR, ECS, or EKS cluster.

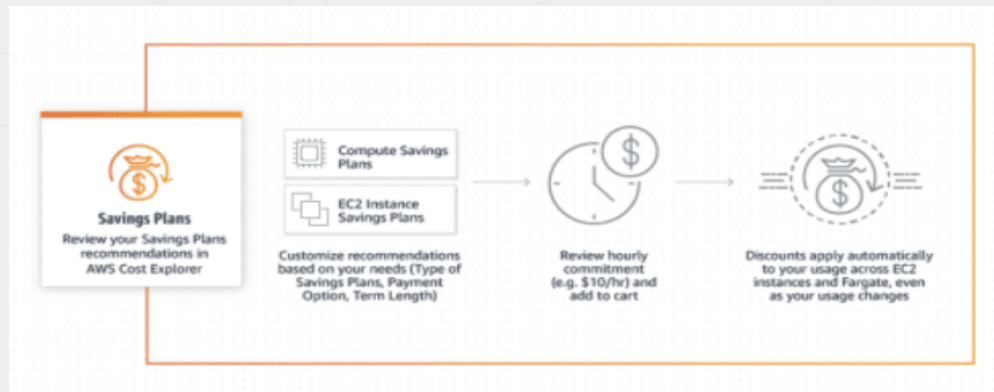
For example, you can shift from C4 to M5 instances, shift a workload from EU (Ireland) to EU (London), or migrate from EC2 to Fargate or Lambda at any time. None of these actions interrupt your AWS Savings Plan's coverage or pricing.

2. EC2 Instance Savings Plans

EC2 Instance Savings Plans apply to a specific instance family within a region and provide the largest discounts, up to 72% (similar to Standard RIs). This plan covers the usage of all instance types, regardless of size, within the same region for that chosen family (e.g., M5 usage in N. Virginia).

The value of this offering lies in its flexibility to switch between sizes, operating systems, and tenancy for like-instances without interrupting your AWS Savings Plan agreement and pricing. For example, you can move from a c5.xlarge running Windows to a c5.2xlarge running Linux without making any changes to your savings plan.

How AWS Savings Plans Work



AWS Savings Plans vs. Reserved Instances

The following section highlights the key differences between traditional Reserved Instances and their new, more flexible counterpart, AWS Savings Plans.

Offering Types

Unit	AWS EBS Type	AWS Savings Plans
Standard RIs and Convertible RIs	Standard RIs and Convertible RIs	EC2 Instance Savings Plans and Compute Savings Plans
Commitment	One year or three years	One year or three years
Payment	All upfront, partial upfront, and no upfront	All upfront, partial upfront, and no upfront

Discount & Coverage Comparisons

Pricing Category	Standard RIs	Convertible RIs	EC2 Instance Savings Plans	Compute Savings Plans
Savings over On-Demand	Up to 72%	Up to 66%	Minutes	Up to 66%
Lower price for monetary commitment	No	No	Yes	Yes
Applies to any family	No	No	No	Yes
Applies to any size	No	No	Yes	Yes
Applies to any tenancy or OS	No	No	Yes	Yes
Applies to Fargate & Lambda usage	No	No	No	Yes
Applies across any AWS region	No	No	No	Yes
Offers 1- and 3-year terms	Yes	Yes	Yes	Yes

Feature Comparisons

Type of Feature	Standard RI	Convertible RI	EC2 Instance Savings Plan	Compute Savings Plan
Scope	Zonal (for Zonal RI) and Regional (for Regional RI)	Zonal (for Zonal RI) and Regional (for Regional RI)	Global	Global
Estimated cost reduction	Up to 72%	Up to 66%	Up to 72%	Up to 66%
Average discount for 1 year	38%	29%	38%	29%
Average discount for 3 years	58%	51%	58%	51%
Availability of 1 year tenure	Yes	Yes	Yes	Yes
Availability of 3 years tenure	Yes	Yes	Yes	Yes
Instance family flexibility	No	Yes (conditions apply)	No	Yes
Operating system flexibility	No	Yes	Yes	Yes
Regional flexibility	No	No	No	Yes (except China)
Tenancy flexibility	No	No	Yes	Yes
Payment options	All, Partial, No Upfront	All, Partial, No Upfront	All, Partial, No Upfront	All, Partial, No Upfront
Availability of capacity reservation	Yes	Yes	No	No
Applicability of account limits	Yes	Yes	No	No
Entitled to sell in the marketplace	Yes	No	No	No
Instance size flexibility	Yes, for Linux OS	Yes, for Linux OS using Exchange Reserved Instances API and console	Yes	Yes
Availability Zone flexibility	Yes, for Regional RI Linux OS using Modify Reserved Instances API and console	Yes, for Regional RI Linux OS using Exchange Reserved Instances API and console	Yes	Yes
List of Supported AWS services	EC2, RDS, Elastic Cache, DynamoDB, Redshift	EC2	EC2	EC2, Fargate, Lambda
Offload/Exchange Commitment	Sell under AWS Marketplace	Exchange for another instance size, family, OS, tenancy, term, or payment option	None	None

Term Comparisons

Unit	Reserved Instance	EC2 Instance Savings Plan	Compute Savings Plan
Average 1 Year Discount	38%	29%	29%
Average 3 Year Discount	58%	58%	51%
Instance Family	Fixed	Fixed	Flexible
Instance Size	Fixed (except Linux)	Flexible	Flexible
Geography	1 Region	1 Region	Flexible
OS	Fixed	Flexible	Flexible
Service	EC2 / RDS	EC2	EC2 / Fargate

Standard RI Flexibility Comparisons

Type	Standard Regional RIs	Standard Zonal RIs	EC2 Instances Savings plans
Commitment Unit	Per instance	Per instance	Per Dollar
Geography	Region specific	Zone specific	Region specific
Instance Family	Fixed	Fixed	Fixed
Tenancy	Fixed	Fixed	Any tenancy
Operating System	Fixed	Fixed	Any OS

Convertible RI Flexibility Comparisons

Type	Convertible Regional RIs	Convertible Zonal RIs	AWS Compute Savings plans
Commitment Unit	Per instance	Per instance	Per Dollar
Geography	Region specific	Zone specific	Any region
Instance Family	Exchangeable	Exchangeable	Any family
Tenancy	Exchangeable	Exchangeable	Any tenancy
Operating System	Exchangeable	Exchangeable	Any OS

Reserved Instance and AWS Savings Plan Pricing

The following tables demonstrate savings for an M5 12xlarge EC2 instance across 1- and 3-year terms for Standard RIs, Convertible RIs, and AWS Savings Plans.

1 Year Standard Reserved Instance / 1 Year EC2 Instance Savings Plan

Payment Option	Upfront	Monthly	Effective Hourly	Savings over On-Demand
No Upfront	\$0.00	\$1,074.56	\$1.4720	36%
Partial Upfront	\$6,140.00	\$511.73	\$1.4020	39%
All Upfront	\$12,035.00	\$0.00	\$1.3740	40%

1 Year Convertible Reserved Instance / 1 Year Compute Savings Plan

Payment Option	Upfront	Monthly	Effective Hourly	Savings over On-Demand
No Upfront	\$0.00	\$1,235.89	\$1.6930	27%
Partial Upfront	\$7,061.00	\$588.38	\$1.6120	30%
All Upfront	\$13,840.00	\$0.00	\$1.5800	31%

3 Year Standard Reserved Instance / 3 Year EC2 Instance Savings Plan

Payment Option	Upfront	Monthly	Effective Hourly	Savings over On-Demand
No Upfront	\$0.00	\$741.68	\$1.0160	56%
Partial Upfront	\$12,362.00	\$343.10	\$0.9400	59%
All Upfront	\$23,241.00	\$0.00	\$0.8840	62%

3 Year Convertible Reserved Instance / 3 Year Compute Savings Plan

Payment Option	Upfront	Monthly	Effective Hourly	Savings over On-Demand
No Upfront	\$0.00	\$852.64	\$1.1680	49%
Partial Upfront	\$14,216.00	\$394.93	\$1.0820	53%
All Upfront	\$27,864.00	\$0.00	\$1.0600	54%

How to Purchase an AWS Savings Plan

1. Review your AWS Savings Plan recommendations for your account.
2. Consider your existing infrastructure needs (instance families, regional requirements) and any active RI commitments.
3. Right-size your existing infrastructure if applicable.
4. Determine what level of AWS Plan commitments you are comfortable with.
5. Purchase your AWS Savings Plan commitments through the Purchase Savings Plan page.
6. Monitor your commitments post-purchase to ensure maximum use while still avoiding On-Demand overages.

Understanding AWS Savings Plan Recommendations & Cost

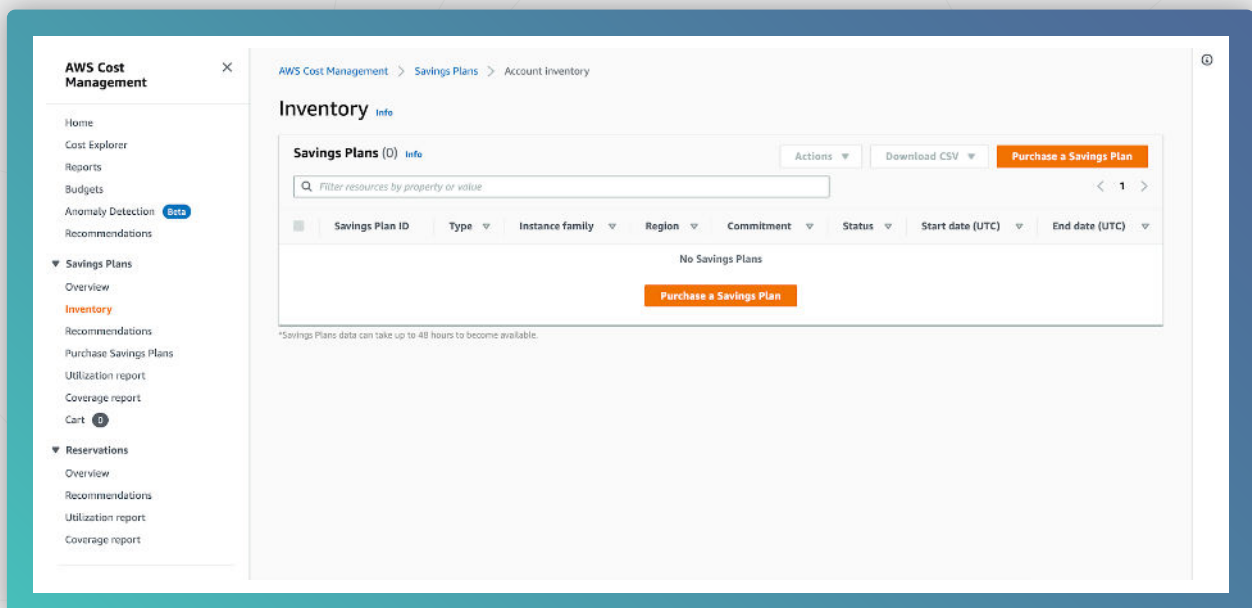
You can view the recommendations for your AWS account through the console. There are three primary metrics to be aware of:

- **Monthly On-Demand Spend:** An estimated monthly cost of total usage without discounts over a selected period (including all active Savings Plans). This metric is used to illustrate effectiveness of your existing Savings Plans versus On-Demand usage.
- **Estimated Monthly Spend:** A monthly cost projection based on current commitment recommendations. This metric includes any forecasted On-Demand usage due to hour-to-hour variations.
- **Estimated Monthly Savings:** An estimated net savings for a selected period based on the current (unpurchased) commitment recommendations.

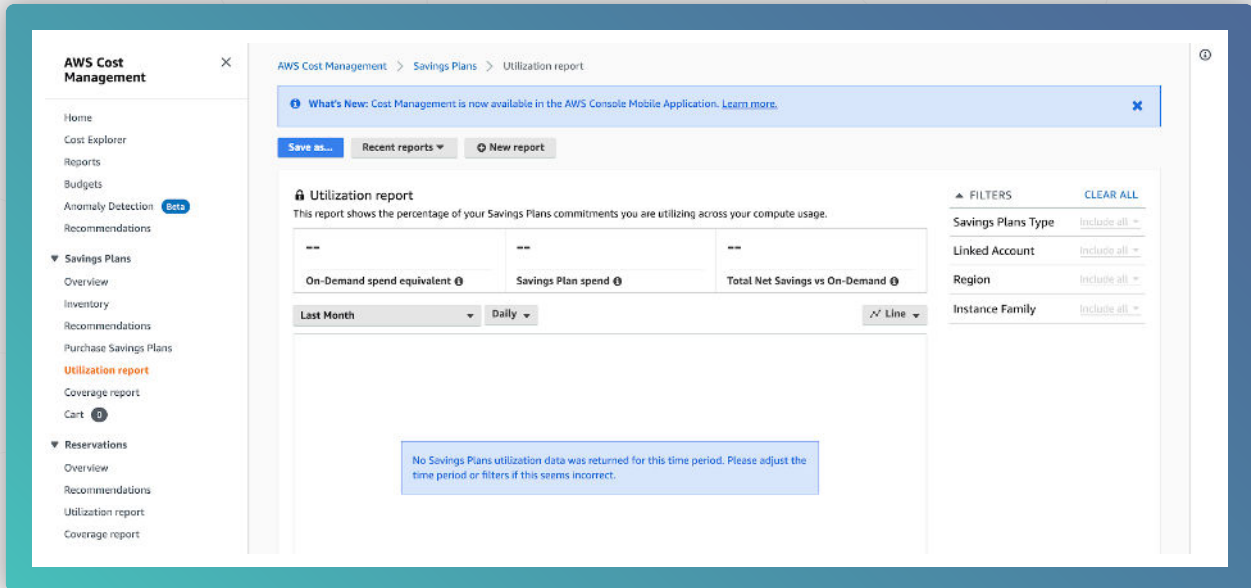
Monitoring AWS Savings Plans

Monitoring your usage is an essential part of [managing your AWS Savings Plans](#). This helps you understand how discounts are applied and what usage types are covered under your Savings Plans. You can manage usage applicable to your Savings Plans in many ways, such as:

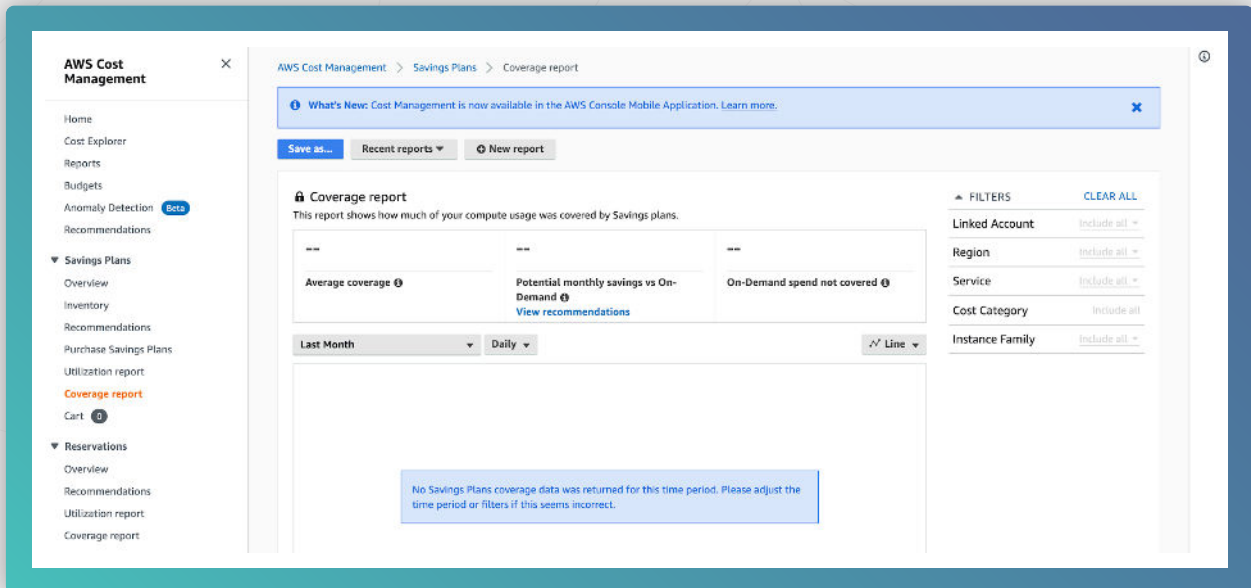
Via the Inventory: Shows a detailed overview of the Savings Plans you own or have queued for future purchases.



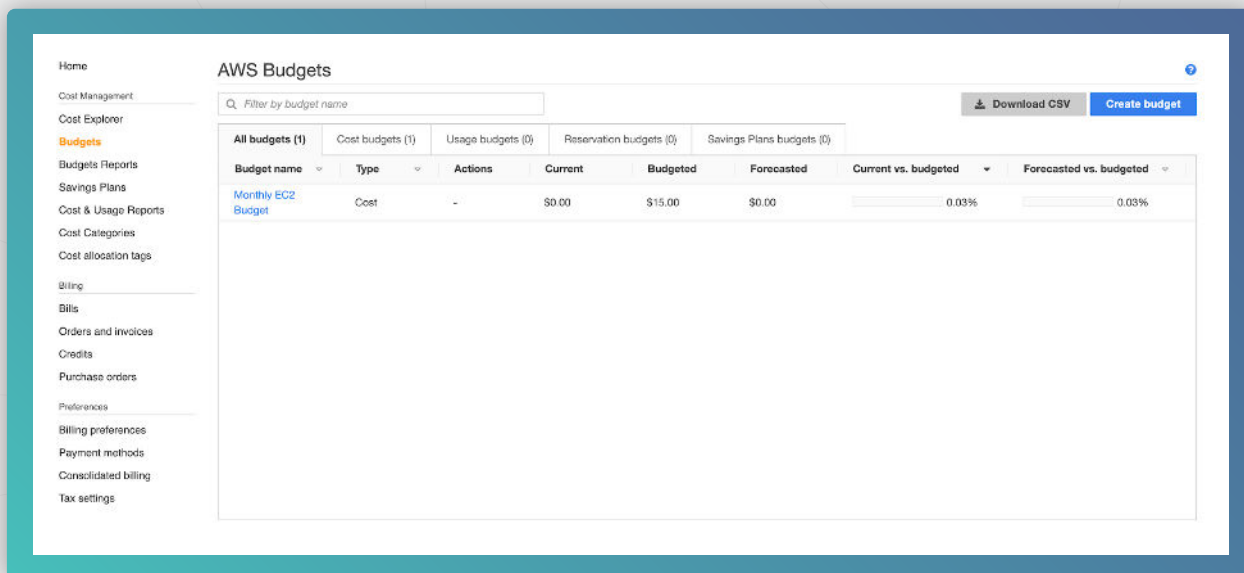
Via the Utilization report: Shows you the percentage of your Savings Plans commitment that you're using across your computer usage.



Via the Coverage report: Shows how much of your eligible spend was covered by your Savings Plans for a defined period.



Via Budgets: Set budgets for your Savings Plan utilization, coverage, and costs.



The screenshot shows the AWS Budgets console interface. At the top, there's a search bar for budget names and buttons for 'Download CSV' and 'Create budget'. Below this, there are tabs for 'All budgets (1)', 'Cost budgets (1)', 'Usage budgets (0)', 'Reservation budgets (0)', and 'Savings Plans budgets (0)'. The main table displays the following data:

Budget name	Type	Actions	Current	Budgeted	Forecasted	Current vs. budgeted	Forecasted vs. budgeted
Monthly EC2 Budget	Cost	-	\$0.00	\$15.00	\$0.00	0.03%	0.03%

Advantages of AWS Savings Plans

- AWS Savings Plans don't lock you into certain instance types like RIs, which means you can make changes within an instance family to be more cost-efficient.
- AWS Savings Plans can benefit from price changes that occur during your commitment, unlike RIs.
- AWS Savings Plans provide discounts without committing you to any specific instance type.
- AWS Savings Plans can be applied to Fargate as well as EC2 (but not RDS).
- AWS Savings Plans avoid the complexity of convertible RI exchanges.
- AWS Savings Plans require less infrastructure planning.
- AWS Savings Plans allow you to flexibly transfer workloads between instance types, sizes, and generations to meet changing demand and architecture.

Limitations of AWS Saving Plans

- AWS Savings Plans can't be purchased for RDS, Redshift, and other services.
- AWS Savings Plans don't offer reseller opportunities to offload underutilized commitments.
- AWS Savings Plans charge On-Demand prices for overages.
- AWS Savings Plans don't provide capacity reservations.
- AWS Savings Plans don't often provide better discounts compared to RIs.

Advantage of AWS Reserved Instances over AWS Savings Plans

- RIs for shorter terms can be purchased on the marketplace.
- RIs can include discounts for RDS, as well as EC2 (but not Fargate).
- RIs can provide some of the largest discounts on the higher-end (60% or more) in the case of some 3-year upfront terms.

Using Reserved Instances and AWS Savings Plans Together

Purchasing an AWS Savings Plan doesn't free you from following cost optimization strategies. Many consumers who plan on using AWS Savings Plan have already purchased Resource Instances for lower prices. Trying to abandon your RIs by selling them all and buying a Savings Plan isn't an ideal solution for your budget; conversely, having a portfolio of just Reserved Instances often lacks the flexibility your business needs. Using both models balances more significant discounts with long-term flexibility.

By using RIs as a layer on top of your AWS Savings Plan, you have a backup discount program that can reduce cloud costs until the expiration of your reservations. If you have predictable resources that aren't covered by RIs, applying an AWS Savings Plan on them can be a safe option.

Chapter 5: AWS Cost Allocation Tags

Cost Allocation Tags are a feature provided by AWS to help you closely monitor the usage and costs of your AWS resources. These tags can be assigned to resources within your AWS accounts, allowing you to construct a customized cost allocation and governance framework that best fits your organization.

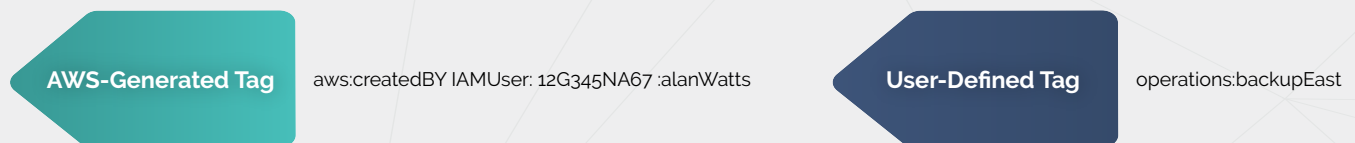
Cost Allocation Tags are helpful in better organizing the costs and usage of your AWS resources, however, they are best used in conjunction with other cost analysis tools such as [Cost Explorer](#), [AWS Budgets](#), [Budget Reports](#), [Cost and Usage Reports \(CUR\)](#), [Cost Categories](#), and [Anomaly Detection](#).

What are AWS Cost Allocation Tags?

AWS Cost Allocation Tags are labels that you can assign to any applicable AWS resource. Each tag consists of two things: a **key** and a unique **value**. AWS requires that all user-defined cost allocation tags use unique keys; for AWS-generated cost allocation tags, the tags are defined, created, and applied by AWS itself or AWS Marketplace ISV using more universal conventions.

Types of AWS Cost Allocation Tags

There are two types of AWS cost allocation tags, as mentioned earlier: AWS-generated and user-defined cost allocation tags. Both types of tags require activation via the Billing and Cost Management Console to serve as cost allocation tags vs. regular operational tags (further explained in a section below). Also, applied tags are visible for only the instances that are provisioned after the activation of tags.



The two types of tags: AWS-Generated and User-Defined

AWS-Generated Cost Allocation Tags

AWS-generated cost allocation tags are viewable in the Billing and Cost Management Console and in reports. They cannot be found elsewhere, such as the AWS Tag Editor. Tags generated by AWS use the same key: `aws:createdBy` which is designed to capture all relevant information related to the creation of a new resource in AWS for future reference. For example::

- `aws:createdBy = IAMUser: ASGHNBBDGTRE :userName`
- `aws:createdBy = AssumedRole: NBVCXGFD :userRole`

Once AWS-generated cost allocation tags are activated by the account manager, the tags are automatically activated for all account members.

Examples of AWS-Generated Tag Values

- **account-type:** States the account type (e.g., *FederatedUser*, *AssumedRole*, *IAMUser*, and *Root*)
- **account -id:** tracks the total number of root accounts or federated users who are responsible for creating resources
- **access-key:** tracks the IAM access key used and the session role name if applicable
- **user-name:** records a user name when available

There are a few limitations to AWS-Generated Cost Allocation Tags; one such limitation is these tags cannot be applied to all AWS services. The services in which these tags can be applied include:

- AWS CloudFormation
- Amazon ElasticCache
- Amazon S3 Glacier
- AWS Data Pipeline
- AWS Elastic Beanstalk
- Amazon Kinesis
- Amazon EC2
- Elastic Load Balancing
- Amazon Relational Database Service

Considerations for AWS-Generated Cost Allocation Tags

- Can only be activated by a management account
- You can't update, edit or delete the tags
- Cannot apply to the resources that were created before the tag was activated
- The maximum active tag keys for Billing and Cost Management reports is 500
- Created using CloudTrail logs, so AWS Generated Tag creation can fail if the log file is too large to accept new entries
- Tag names and values are automatically assigned
- Tag names don't count towards the user-defined resource tag limit of 50
- Null tag values will not appear in Cost Explorer and AWS Budgets
- If there is only one tag value that is also null, the tag key will also not appear in Cost Explorer or AWS Budgets

User-Defined Cost Allocation Tags

User-defined cost allocation tags are defined, created, and applied to resources by members of your AWS account. Usage of these tags follow the same process as AWS-generated tags, that is, they require activation and are applied only to the resources that are initiated after the activation process. These kinds of tags allow for level-2 tagging, meaning that you can tag resources launched by other resources (e.g., an Amazon EMR's EC2).

User-defined cost allocation tags can be viewed via the Cost Management Console after enabling Budgets, Cost Explorer, Legacy reports, or AWS Cost and Usage Reports. User-defined tags also appear in the cost allocation report for you to manage and track your AWS costs.

How to Create User-Defined Cost Allocation Tags

There are several ways for you to create user-defined cost allocation tags, including:

- AWS Tag Editor
- AWS Management Console
- APIs

Considerations for User-Defined Cost Allocation Tags

- The reserved prefix is "user:" which is displayed in the Cost Allocation Report
- Each key can only be used once for each resource
- In some services, you can tag a resource when you create it
- You can't backdate the application of a tag
- Tags only start appearing on your Cost Allocation Report after you apply them
- You can apply standard base-64 encoding to your tag
- Billing and Cost Management does not encode or decode your tag for you
- Tags on non-metered services can be activated; however, these tags do not populate in the Cost Management suite

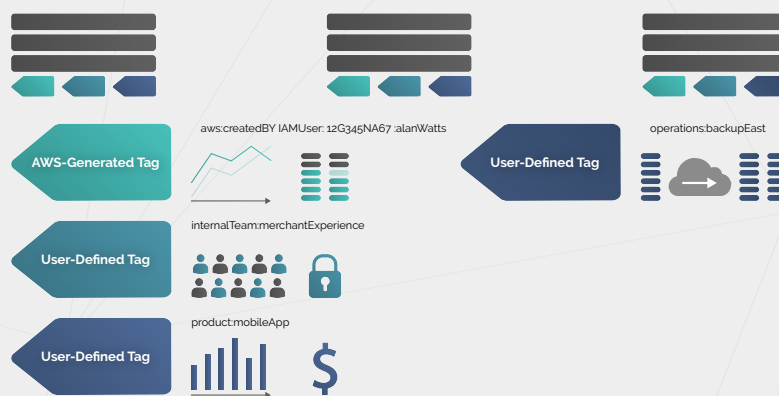
Activating Cost Allocation Tags

Cost Allocation Tags must be activated from a list of regular AWS tags. In other words, you must first create an AWS tag and then instruct AWS to use it specifically as a Cost Allocation Tag. This approach helps AWS lower the indexing overhead in its billing logs unless requested by the account administrators. The instructions to activate user-defined are documented in [this article](#), and the AWS-defined tags are defined in [this article](#), even though they involve identical steps.

AWS Cost Allocation Tag Use Cases

Some common use cases for cost allocation tags include:

- Automating operations
- Managing access and minimizing security risk management
- Identifying costs for business units, departments, projects, products, or regions
- Identify underutilized resources
- Building disaster recovery plans
- Forecasting resource needs



Your AWS resources can be tagged to designate the owner, the product-line, or operations team

Best Practices for Using AWS Cost Allocation Tags

This section is a high-level overview of best practices you should keep in mind when using AWS Cost Allocation Tags. Use these best practices to quickly design a tagging strategy that has room to evolve with your organization's needs over time.

- **Identifying Tag Requirements:**
 - Employ a cross-functional team to identify tag requirements
 - Assign owners to define consistent tag values for each key
 - Focus on required and conditionally required tags
 - Start small; less is more
- **Naming Tags and Resources:**
 - Design and enforce a consistent tag nomenclature
 - Adhere to consistent notation formats for different resource types
- **Cost Allocation Tags:**
 - Align Cost Allocation Tags with financial reporting dimensions
 - Use them along with Linked Accounts to associate an AWS account with a master payer account
 - Avoid multi-valued Cost Allocation
 - Tags Tag everything
- **Tag Governance and Data Management:**
 - Integrate with authoritative data sources
 - Use compound tag values (where one key has multiple values) judiciously
 - Use automation such as CloudFormation templates to proactively tag resources
 - Constrain tag values with [AWS Service Catalog](#)
 - Propagate tag values across related resources (for example all EBS volumes attached to an EC2)
 - Lockdown tags used for access control (where tags are used to limit security access privileges to specific resources)
 - Remediate untagged resources by periodically creating a list
 - Implement a tag governance body that meets regularly to review best practices

AWS Billing

Cost Allocation Tags appear in the form of a report as comma-separated values in a CSV file that can be generated via the AWS Billing and Cost Management Console. The usage and cost of resources are grouped within the CSV file based on your active tags. This report is known as the [Cost Allocation Report](#).

Cost Allocation Tags in the Billing Dashboard

Via the billing dashboard, you can access the cost allocation report and apply tags representing your business categories such as cost centers, owners, application names, production, testing, and so on.

You can view both tagged and untagged resources and organize them accordingly to better calculate the charges for your resources. For example, if you tag your AWS resource with the application name then you will have the ability to track even a single application that utilizes the tagged resource.

AWS Cost Allocation Reports

A [Cost Allocation Report](#) is a list of the usage of your AWS resources grouped by your product category and any linked account user. It also contains, like detailed Cost and Usage Reports, the same line items with the addition of your Auto-Generated and User-Defined Cost Allocation Tags.

There are two types of cost allocation reports:

- Monthly Cost Allocation Report
- Hourly Cost Allocation Report

During a billing cycle, AWS first generates an estimated cost allocation report that gets overwritten by the actual usage throughout the billing period.

Viewing a Cost Allocation Report

You can download a CSV file and view it in a desktop spreadsheet application. The data in the file can then be organized by creating a pivot table to group the keys and values to get the combined values of tagged resources.

Unallocated Resources in Cost Allocation Reports

Unallocated resources by default follow that standard billing aggregation, that is, they get organized by the line item, account, and product.

Unexpected Costs Associated with Tagged Resources

There are several possible situations for the unexpected appearance of costs associated with tagged resources, these situations include:

- Usage exceeds the AWS Free Tier
- AWS Free Tier expires
- Bill is received after account closure
- Disabled regions with resources still active for those regions
- Elastic Beanstalk environments terminated before the termination of resources
- Elastic Load Balancing (ELB) termination before the termination of EC2 instance
- Services started in AWS OpsWorks
- Undeleted Amazon Elastic Block Store volumes and snapshots
- Unreleased Elastic IP addresses
- Services launched by other services
- Costs associated with storage services such as RDS and S3

Managing Tags Programmatically

There are several ways to automate the tagging process. Here are some:

- Auto Scaling Groups (ASG) are designed to scale in and out the number of nodes in a cluster based on application workload. They are also able to automatically tag the nodes that are launched.
- In 2017, [AWS unveiled AWS Tagging APIs](#) that can be accessed programmatically via a script, or as part of a continuous delivery process. You can find the API functions [in this article](#).
- The AWS Command Line Interface (CLI) is a popular method to apply changes to your AWS environment by typing commands and avoiding the need to click in a UI. The AWS CLI supports the tagging API functions.

Set Tag-based KPIs

The natural evolution of tagging and measuring is benchmarking. Defining KPIs, or Key Performance Indicators, is a great way to understand the state of your resource usage visibility over time. Here are a few examples:

- Percentage of resources where tagged resource has no tag value
- Percentage of resources that have no tags at all
- Percentage of un-tagged resources trending over time

Read [6 Cost KPIs to Drive Hybrid Cloud Value](#) for ideas on how to measure your cloud optimization progress. Here are three additional KPIs that you can add to the list for tracking tagging success:

Percentage of untagged resources
Percentage of resources tagged with null value
Trend graph of your tagging coverage over time

How to Get the Most Out of AWS Cost Allocation Tags

Adding AWS Cost Allocation Tags to your resources is an excellent way of organizing AWS resources and separating costs, but don't stop there. Tags should be used in conjunction with other cost analysis tools. Here are some examples of how to make use of your tags:

- Analyze your costs with Cost Explorer
- Manage your costs with AWS Budgets
- Report your budget metrics with budget reports
- Manage costs with [AWS Cost Categories](#)
- Use the AWS Price List API

AWS Cost Management Features

AWS provides many cost management services. The following services are divided into groups with specific capabilities and use cases.

- **AWS Cost Allocation Tags** and **AWS Cost Categories** allow you to organize and construct a cost allocation and governance foundation. They also allow you to use your own tagging strategy.
- **AWS Cost Explorer** and **AWS Cost and Usage Reports** allow you to generate reports to raise awareness and offer accountability for your cloud expenditure. They do so by providing you with detailed allocable cost data.
- **AWS Consolidated Billing**, **AWS Purchase Order Management**, and **AWS Credit** allow you to access and track billing information across the entire organization.

AWS Budget Features

AWS Budgets, AWS Budget Actions, and AWS Service Catalog allow you to budget and keep your spending in check via the budget thresholds, auto alerts, and notifications.

AWS Billing Alarm Features

You can use AWS Budgets and AWS Budget Actions to set alarms, auto alerts, and notifications for overage on any resource that is being used. You have the option to receive your budget alerts for your resources in Amazon Chime, and in Slack using the AWS Chatbot.

AWS Trusted Advisor

Using a trusted advisor is an excellent option for cost optimization. [Trusted Advisor checks](#), across all AWS regions, your AWS infrastructure and creates a summary of the results. With the cost optimization option in the Trusted Advisor, you can view the potential monthly saving, recommendations, and cost optimization checks.

Trusted Advisor can perform the following check categories:

- Potential Monthly Savings
- Cost Optimization Checks for Underutilized Resources
- Cost Optimization Checks for Reservation
- Amazon Route 53 Latency Resource Set Check

Chapter 6: EC2 Spot Instances

What are Spot Instances?

An **AWS EC2 Spot Instance** is an unused EC2 instance which is available for less than the On-Demand price. Spot instances are up to 90% cheaper than On-Demand instances, which can significantly reduce your EC2 costs. A Spot Price is the hourly rate for a Spot instance. AWS sets the Spot price for each instance type in each availability zone based on the evolving supply and demand for Spot instances. Spot instances are cost-effective when you can be flexible with your application's availability and when your applications can be interrupted after a two-minute warning notification.

Spot instances are ideal for stateless, error-tolerant, or flexible applications like data analysis, batch jobs, background processing, and optional tasks. These instances are closely integrated with AWS services like Auto Scaling, EMR, ECS, CloudFormation, Data Pipeline, and AWS Batch. You can easily combine Spot instances with On-Demand, RI, and Savings Plans instances to optimize workload costs and performance.

Reasons to Use Spot Instances

- Low prices
- Massive scale
- Easy to use
- Easy to automate
- Supports other AWS services

Spot Instances VS On-Demand Instances

Metrics	Spot Instances	On-Demand Instances
Launch time	Launches instantly if the Spot Request is active and capacity is available.	Launches instantly when you make a manual launch request and capacity is available.
Available capacity	Delivers launch requests until capacity becomes available.	Sends an insufficient capacity error when the request is made and no capacity is available.
Hourly price	Varies depending on supply and demand.	Remains the same.
Rebalance recommendation	Sends a warning signal when the instance is at high risk for interruption.	Continues to run until you terminate or hibernate the instance.
Instance interruption	Interruptible by AWS EC2 when capacity is no longer available, the prices exceed your budgeted max rate, or the demand for Spot instances increases.	Remains uninterrupted until you terminate or hibernate the instance.

Strategies for Using Spot Instances

There are 2 key strategies for using Spot Instances:

1. Maintain a minimum number of compute resources by launching a core group of On-Demand Instances and supplementing them with Spot Instances when required.
2. Launch Spot instances with a fixed duration, called Spot blocks, which are designed to be uninterrupted and run continuously for the duration you choose.

Spot Instance Best Practices

- **Prepare individual instances for interruptions:** The best way to be fault-tolerant is to handle Spot instance interruptions with smooth workload rebalancing. Use EC2 instance rebalance recommendations and Spot Instance interruption notices to avoid feeling any hard cutoffs.
- **Be flexible in terms of instance types and availability zones:** A Spot instance pool consists of unused EC2 instances with the same instance type and availability zone. You should be flexible about which instance types you request and in which Availability Zones. Read our article in this series related to [AWS Data Transfer Pricing](#) to consider the additional cost of data transfers.
- **Use EC2 Auto Scaling groups or Spot Fleet to manage your total capacity:** Spot allows total capacity with vCPUs, storage, storage or network throughput. Auto Scaling groups and Spot Fleet allow you to start and maintain a target capacity and automatically request resources.
- **Use the capacity-optimized allocation strategy:** Allocation strategies in auto-scaling groups help you deploy your target capacity without manually searching for the Spot instance pools with free capacity.
- **Use proactive capacity balancing:** Capacity rebalancing helps maintain availability by adding a new Spot instance to the fleet before a running Spot instance receives the two-minute notification. It balances the capacity-optimized allocation strategy and the policy of mixed entities.
- **Use integrated AWS services to manage your Spot instances:** Other AWS services like EMR, ECS, AWS Batch, EKS, SageMaker, AWS Elastic Beanstalk, and GameLift integrate with Spot to reduce total invoice costs without having to manage individual instances or fleets.

Deploying Spot Instances

A Spot Instance can be started in 2 different ways: you can either create a Spot Instance request or have AWS EC2 create one on your behalf. When starting a Spot instance, it is important to know that Spot pools have their own rates and these rates change less often. AWS will give you a 2-minute warning before taking back any Spot capacity you are using.

Request Methods

Requests can be made using any of the following:

- Spot Instance requests
- Spot Fleet requests
- EC2 Fleet requests

Launch Locations

Spot Instances can be launched in 3 different places in AWS:

- Within a [launch group](#).
- Within an [Availability Zone group](#).
- Within a [Virtual Private Cloud \(VPC\)](#).



The Amazon EC2 Spot Instances launch steps (Source: <https://aws.amazon.com/>)

Spot Fleets

A [Spot fleet](#) is a collection of Spot instances (note that this collection can optionally contain On-Demand instances as well). The target capacity you specify in the Spot Fleet request defines the number of Spot instances and On-Demand instances. Requests for Spot instances are fulfilled whenever 1) spare capacity is available and 2) the maximum rate specified in your request exceeds the current Spot rate. Spot Fleets strive to maintain their target capacity, meaning they will launch new spot instances to regain capacity if fleet members are terminated. You can set a maximum hourly rate for your fleet and Spot Fleet will continue launch instances until that amount has been met. When your hourly rate is met, the Spot Fleet stops starting instances even if the target capacity has not been reached.

Key Features of Spot Fleets

- Amazon EC2 auto scaling integration
- Optimized for cost
- Reduce the likelihood of interruptions
- On-Demand run instances function integration
- Stop/Hibernate and resume workloads
- Track when Spot instances run and terminate
- Amazon EMR integration
- Amazon CloudFormation integration
- Amazon ECS integration
- Amazon Batch integration
- Attach encrypted EBS volumes at launch
- Control your Spot Instance budget
- Support for Third Party Integration

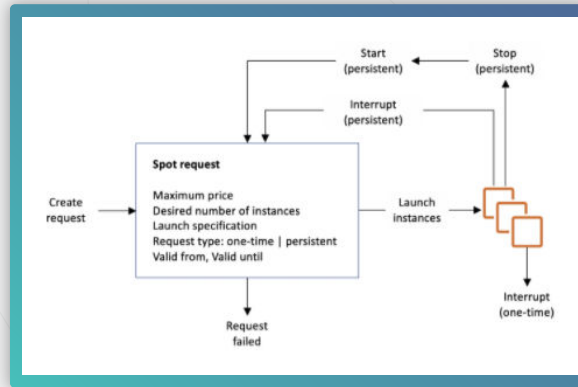
Important Spot Instance Concepts

Spot Instance Request

A [Spot Instance Request](#) defines the number of instances, instance types, availability zones, and the maximum hourly rate you are willing to pay per instance. As soon as the maximum price exceeds the current Spot price and capacity is available, EC2 instances will immediately respond to the request. Otherwise, EC2 instances will wait until the request can be fulfilled or until the request is cancelled. The one-time or persistent request type determines whether the request is reopened when EC2 interrupts a Spot instance or when you stop a Spot instance. If the request is persistent, the request is opened again after your Spot Instance is interrupted. If the request is not persistent and you stop your Spot instance, the request only opens after you start your Spot Instance.

Spot Instance Interruptions

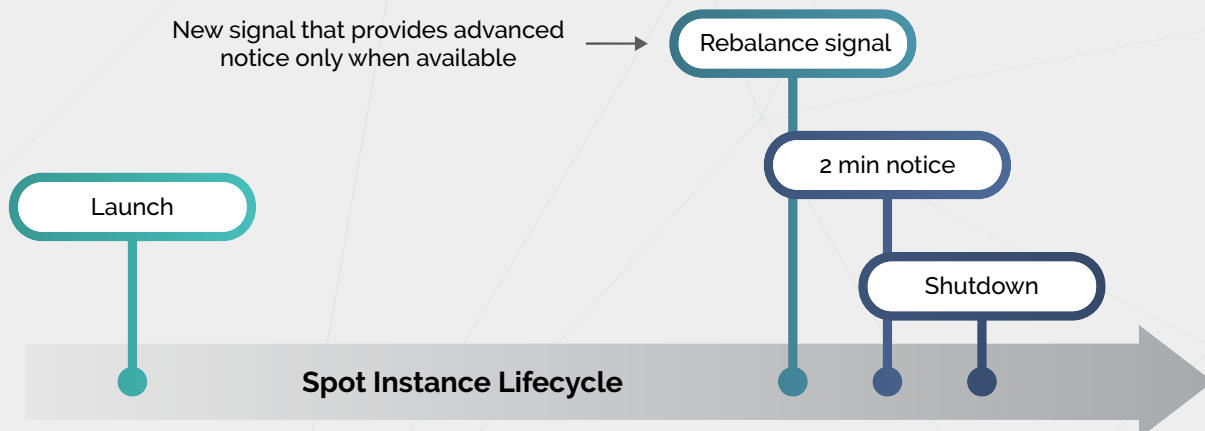
[Spot Instance interruption](#) is when EC2 instances reclaim a Spot instance's capacity to meet capacity commitments for reservations or On-Demand use. The demand for Spot instances can vary considerably from moment to moment, and the availability of Spot instances depends on how many unused EC2 instances are available. There is a possibility that your Spot instance will be interrupted, so you need to make sure that your application is prepared to handle an interruption.



The state flow of a spot instance's requests and interruptions (Source: <https://aws.amazon.com/>)

Rebalance Recommendations

An EC2 [Instance Rebalance Recommendations](#) is a signal that notifies you when a Spot Instance is at high risk of interruption. The notification comes earlier than the two-minute interruption of the Spot instance, giving you the ability to manage the Spot instance proactively. This allows you to rebalance your workload across new or existing Spot instances that are not at a high risk of interruption.



When possible, the rebalance signal provides additional time to take a graceful shutdown action

Spot Instance Advisor

The [Spot Instance Advisor](#) helps you identify pools with the least probability of disruption and offers the same savings you get over On-Demand rates. When selecting a Spot instance, you should account for the interruption tolerance of your application and your cost-saving goals. The lower the interrupt rate, the longer your Spot instances will likely run.

Spot Instance Limits

[Spot Instance Limits](#) are a limit on the number of ongoing and requested Spot instances per AWS account per region. These limits are defined by active and requested (pending) vCPU count. If you terminate your Spot instance but do not cancel the Spot instance request, the number of requests will not match your Spot instance vCPU limit until Amazon EC2 detects the Spot instance cancellations and closes the requests.

Burstable Performance Instances

If you start your Spot instances with a [burstable performance instance](#) type and plan to use them immediately and for a brief period without enough time for accruing CPU credits, it is recommended to run them in default mode to avoid higher costs. If you launch burstable performance Spot Instances in Unlimited Mode and burst the CPU immediately, you'll spend surplus credits for bursting. If you use the instance for a short time, the instance will have no time to amass CPU credits to repay the excess credits, and you will be charged for the excess credits when you terminate the instance.

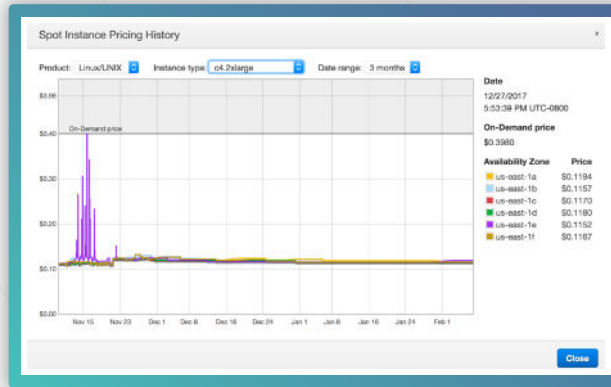
Cost Saving Use Cases with AWS Spot Instances

- 1. Big Data & Analytics:** Big data, machine learning, and Natural Language Processing (NLP) workloads can be fast-tracked with Spot Instances. Spot instances provide acceleration, scaling, and deep cost savings for time-critical, hyper-scalable workloads needing rapid data analysis. Spot instances can be used with Amazon EMR, Hadoop, or Spark to handle enormous amounts of data.
- 2. Containerized Workloads:** Container clusters can be with Spot Instances at a fraction of the cost. Containers are stateless, fault-tolerant and ideal for Spot Instances. Spot clusters can be created with AWS ECS or Kubernetes to operate containerized workloads.
- 3. High Performance Computing:** Accelerate large computing tasks like genome sequencing, Computational Fluid Dynamics (CFD), and algorithmic trading by performing huge parallel jobs. Spot instances are integrated with AWS Batch, AWS CloudFormation and other AWS services, providing a complete solution for various large-scale computing workloads.
- 4. Web Services:** Save up to 90% on web services and applications with Spot instances. Deploy an EC2 Spot Fleet behind a load balancer to scale to tens of thousands of instances, serving billions of service requests with Spot Instances.
- 5. CI/CD & Testing:** Jenkins can be configured with the EC2 Spot Plug-In to scale Spot instance fleets based on the number of incomplete jobs that are in queue. You can increase cost savings by using smaller instances for CI, as these processes do not require much power for testing. Many workload types (canary, security testing) can be performed more cheaply via Spot instances.
- 6. Image and Media Rendering:** Media and entertainment studios can cost-effectively manage rendering workloads using Spot instances by scaling on-premises or cloud infrastructures with almost unlimited capacity, as required by projects and timelines.

Pricing and Cost Savings

Spot Instance Pricing model

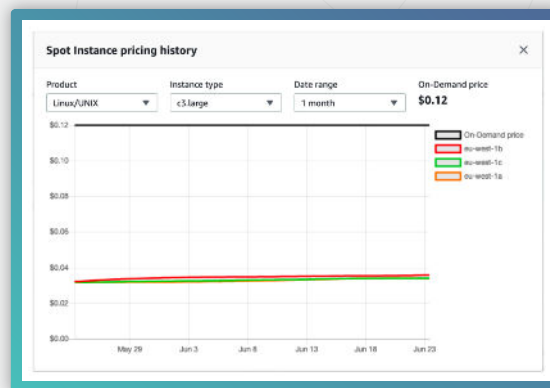
You pay the Spot price only for the period your instances are operating. [Spot Instance prices](#) are set by EC2 and are adjusted based on trends in supply and demand. Spot instances are also available to operate for a predefined duration in hourly increments of up to 6 hours in length.



The historic pricing by type and Availability Zone

Spot Instance Pricing History

It is recommended you use the standard maximum On-Demand price when requesting Spot instances. If you want to specify a maximum price, review the [Spot Instance Pricing History](#) first. You can view the Spot instance pricing history for the last 90 days and filter by instance type, operating system, and availability.



The historic pricing filtered by type and time range

Spot Instance Cost Saving

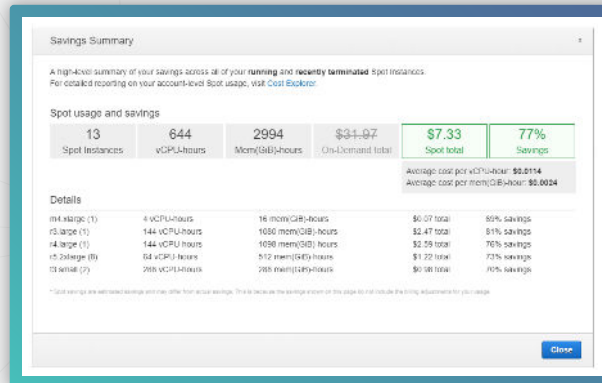
Usage and savings information for fleet-level Spot instances or all ongoing Spot instances can be viewed for [cost-saving](#) purposes. At fleet level, the usage and savings information includes all instances started and terminated by the fleet. You can view this information from as recently as the last hour or up to three days.

Spot usage and savings					
4	266	700	\$9.55	\$2.99	69%
Spot Instances	vCPU-hours	Mem(GiB)-hours	On-Demand total	Spot total	Savings
				\$0.0112	\$0.0043
				Average cost per vCPU-hour	Average cost per mem(GiB)-hour
Details					
t3.medium (1)	2 vCPU hours	4 mem(GiB)-hours	\$0.01 total	70% savings	
m4.large (1)	144 vCPU hours	576 mem(GiB)-hours	\$2.52 total	68% savings	
t2.micro (2)	120 vCPU hours	120 mem(GiB)-hours	\$0.46 total	70% savings	

Your available savings potential

Spot Instance Data Feed

AWS EC2 provides a [Spot Instance Data Feed](#) that describes your Spot Instance use and pricing. This data feed is sent to an Amazon S3 bucket you specify when subscribing to the data feed. Data feed files are sent to the bucket hourly, and each usage hour is typically recorded in a single data file. These files are gzip-compressed before being delivered to your bucket. EC2 may write multiple files for a given usage hour if the data is too large.



Your spot usage and savings summary

Spot Instance Integration with Other AWS Services for Cost Savings

- Amazon EMR Integration:** Operate AWS EMR clusters on Spot Instances to reduce the cost of processing massive amounts of data for Hadoop clusters. Simply mix Spot Instances with On-Demand and Reserved Instances using the instance fleet feature.
- AWS CloudFormation Integration:** Organize and deploy AWS resources using CloudFormation scripts; CloudFormation also lets you describe dependencies and parameters to be passed in at runtime.
- Auto Scaling Integration:** Manage Spot instances using Auto Scaling groups to maintain application availability; you can scale your Spot capacity automatically depending on any conditions or pricing rates you define.
- Amazon ECS Integration:** Operate ECS Clusters on Spot instances to reduce the operational cost of maintaining your containerized applications.
- Amazon Batch Integration:** Dynamically schedule your batch computing workloads for Spot instances to reduce the cost of performing your batch jobs.

Conclusion

Spot instances are an excellent way to lower your AWS spend, however it is important to keep in mind which kinds of applications or workloads best fit the nature of Spot instance usage. Spot Fleets also enable you to minimize the potential risks of using Spot instances when proper workload transition plans are set into place.

Chapter 7: AWS Bill Analysis

10 Ways to Use AWS Bill Analysis Tools

At one point in time, AWS had only one tool for tracking your resource spending: the AWS Detailed Billing Report (DBR) which was a log of billing records. But that report has since been deprecated in favor of the Cost and Usage Report (CUR) along with other bill analysis tools. As the AWS ecosystem continues to grow and evolve every year, new functionalities and APIs have emerged—each with their own value for resource optimization or usage forecasting.

With so many new ways to analyze costs, it can be difficult to know which AWS billing tools to use and when, even before you leverage third-party tools with their own specialized value-add. In this article, we will map out the various cost management tools currently available from AWS and their main use cases.

AWS Bill Analysis Use Cases

In the table below, we have grouped various AWS cost tools by use case. We have associated each use case with a single verb to help simplify the concept in our reader's mind, and we have listed the corresponding AWS tools relevant to each use case.

Use Cases	Capabilities	On-Demand Instances
Access	Track billing information across the organization in a consolidated view.	AWS Consolidated Billing , AWS Purchase Order Management
Control	Establish effective governance mechanisms with the right guardrails in place.	AWS Cost Anomaly Detection , AWS Identity and Access Management , AWS Organizations , AWS Control Tower , AWS Service Catalog
Organize	Construct your cost allocation and governance foundation and tagging strategy.	AWS Cost Allocation Tags , AWS Cost Categories
Purchase	Leverage discount programs based on workload pattern.	AWS Reserved Instances , AWS Savings Plans , AWS Spot Instances , Enterprise Discount Program
Inspect	Stay up-to-date with resource deployment and cost optimization opportunities.	AWS Cost Explorer
Report	Raise awareness and accountability with reporting by cost-center.	AWS Cost Explorer , AWS Cost and Usage Report , Monthly Cost Allocation Report
Forecast	Estimate your resource utilization and spend with forecast dashboards.	AWS Cost Explorer , AWS Budgets
Budget	Keep your spend in check with custom budget thresholds and auto alert notifications.	AWS Budgets , AWS Budget Actions , AWS Service Catalog
Scale	Scale and schedule your services based on expected utilization patterns and needs.	AWS Instance Scheduler , EC2 Auto Scaling , AWS Trusted Advisor
Rightsize	Align your service allocation size to actual workload demand.	AWS Cost Explorer Right Sizing Recommendations , AWS Compute Optimizer

AWS Bill Analysis Best Practices

In this section, we list ten best practices that will help you stay on top of your spending. These best practices have been chosen based on recommendations from practitioners, along with convenient hyperlinks to AWS web pages introducing the relevant AWS tools.

1. Allocate and Group Costs

Use the [AWS Enterprise Billing Console](#) to check your actual usage rates and allocate costs across the accounts in your consolidated billing family. A key feature of the Enterprise Billing Console is the ability to create billing groups; use these billing groups to apply custom pricing plans or analyze your savings by performing a margin analysis on each billing group. You can also configure Cost and Usage Reports (CUR) on each billing group.

[AWS Dashboard Graphs](#) are also a key feature of the AWS Billing and Cost Management console. Dashboard Graphs provide a default view into your spend summary, month-to-date spend by service, and month-to-date top services by spend.

2. Track Estimated Spend

Use [AWS Billing Alarms](#) to monitor your estimated AWS charges. This feature calculates your estimated charges daily using CloudWatch metric data collection. Alerts are sent via email whenever your actual spending exceeds the threshold specified. You can review billing data and receive alerts for each linked account when using consolidated billing, provided the management account has "Receive Billing Alerts" enabled.

Note that AWS Billing Alarms cannot be used for accounts that are part of the Amazon Partner Network (APN).

3. Track Actual Spend

Use [AWS Budgets](#) to define custom budgets that track your cost and usage. You can set up alerts (via email or SNS) on actual or forecasted usage thresholds. For example, you might want to receive an alert when your RI or Savings Plan utilization drops below a certain threshold.

To take action on cost and usage-related events, set up [AWS Budget Actions](#). These actions can be executed either automatically or with your approval.

AWS Budgets integrates with other AWS services, such as AWS Cost Explorer, [AWS Chatbot](#), and AWS Service Catalog.

Use [Budget Reports](#) to monitor your budgets. Budget Reports supports up to 50 reports per account. You can send this report to a maximum 50 email addresses to keep your team and stakeholders informed. Each report sent costs \$0.01.

4. Explore The Past Year's Billing Data

Use the [Cost Explorer](#) to view and analyze up to 12 months of your AWS costs and usage. There are several ways to explore your data within Cost Explorer, including: the main graph, CUR, or the Cost Explorer Reserved Instances (RI) reports. You can also use the Cost Explorer to forecast spending and get recommendations for RI purchases.

Cost Explorer provides several preconfigured views for analyzing your cost trends, making it an excellent starting point for analyzing your AWS bill. While using the Cost Explorer through the AWS Console is free, accessing this data via the API incurs a charge of \$0.01 per request.

5. Receive Cost and Usage Reports

Use [Cost and Usage Reports](#) to see detailed billing information categorized by additional metadata (such as cost allocation tags). After you set up CUR for the first time, the current month's billing data is delivered to your designated S3 bucket. You have the option to receive hourly, daily, or monthly reports broken down by product, resource, or tags. CUR updates this report at least once each day.

6. View Spend by Cost Categories

Use [AWS Cost Categories](#) to group cost and usage data in a way that makes sense for your business. You can map these categories to your data using configurable rules that support logic on metadata such as account, tag, service, charge type, and other cost categories. Cost categories become viewable in AWS Cost Explorer, AWS Budgets, and AWS CUR at the beginning of the following month.

7. Monitor Cost Anomalies

Use [AWS Cost Anomaly Detection](#) to create cost monitors powered by machine learning that detect the root cause of anomalous spending. Cost monitors help segment your spend by account, cost category, cost allocation tag, and more. After setup, you can review any detected anomalies through the dashboard — even if they fall below your defined thresholds.

8. Reduce Resource Waste

Use the [AWS Trusted Advisor](#) to receive real-time resource provisioning recommendations that follow AWS best practices. Trusted Advisor measures service limits by checking your service usage against an 80% threshold. This helps avoid overage fees (over-utilization), especially for resources that handle workloads that experience regular spikes or regular growth. This tool also optimizes costs by removing idle or unused resources from your inventory and by making resource reservations.

9. Discover Price Changes

Use [AWS Price Update Notifications](#) to stay up-to-date on AWS pricing, including: AWS price cuts, new instance launches, and new service announcements. You can receive daily notifications via SNS, which include all price changes applied during a given day.

You can also directly query for AWS service pricing through [the AWS Price List Service API](#) and [the AWS Price List API](#).

10. Review Billing and Cost Logs

Use [CloudTrail](#) to capture billing events as a log across your accounts. Billing and Cost events are recorded for actions taken by users, roles, and AWS services from either the console or via an API.

You can monitor when a linked AWS account has been closed, has changed owners, has redeemed a discount code, and has updated its RI/Savings/Credit sharing settings (via the SetRISharing and SetCreditSharing events). The example below shows the format for the event SetContactAddress.

```
{
  "eventVersion": "1.05",
  "userIdentity": {
    "accountId": "111122223333",
    "accessKeyId": "AIDACKCEVSQ6C2EXAMPLE"
  },
  "eventTime": "2018-05-30T16:44:04Z",
  "eventSource": "billingconsole.amazonaws.com",
  "eventName": "SetContactAddress",
  "awsRegion": "us-east-1",
  "sourceIPAddress": "100.100.10.10",
  "requestParameters": {
    "website": "https://amazon.com",
    "city": "Seattle",
    "postalCode": "98108",
    "fullName": "Jane Doe",
    "districtOrCounty": null,
    "phoneNumber": "206-555-0100",
    "countryCode": "US",
    "addressLine1": "Nowhere Estates",
    "addressLine2": "100 Main Street",
    "company": "AnyCompany",
    "state": "Washington",
    "addressLine3": "Anytown, USA",
    "secondaryPhone": "206-555-0101"
  },
  "responseElements": null,
  "eventID": "5923c499-063e-44ac-80fb-b40examplegf",
  "readOnly": false,
  "eventType": "AwsConsoleAction",
  "recipientAccountId": "1111-2222-3333"
}
```

Conclusion

AWS has introduced many new tools, APIs, and functionalities in recent years. The choices may be overwhelming to newcomers. We recommend adopting an essential subset such as the Cost Explorer and Cost Allocation Tags, and consider third-party vendors who leverage AWS' APIs to offer a simpler user experience in a single pane of glass.

Chapter 8: AWS Sizing Tools

6 AWS Sizing Tools

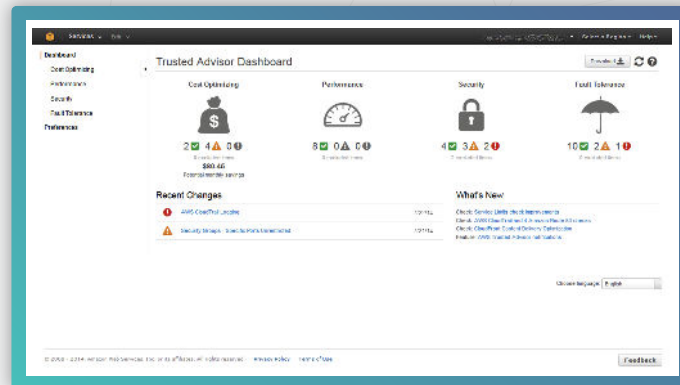
Using AWS, you can set up and deploy all kinds of services within minutes—everything from virtual machines and containers, to file storage, to databases, and even a programming code function. While deploying AWS services can be easy, sizing them can be challenging.

In fact, it's hard to anticipate the amount of resources your new project might require without historical data on a similar application, run with a similar code stack, receiving similar levels of user activity, during peak usage hours. And because of so much unknown, it's far safer to overestimate your resource needs than it is to risk application stability by trying to start off lean. But there is something you can do to dial in those resources soon after deploying them.

In this article, we'll take a look at the most common AWS sizing tools for resource optimization, broken down by their different approaches, so you can start reducing your overall bill.

1. Sizing by Recommendations

Trusted Advisor is an AWS sizing tool that provides resource recommendations across five categories: cost optimization, performance, security, fault-tolerance, and service limits. These recommendations are powered by checks (up to 115 in total) that run in real-time across each category.



Trusted Advisor Dashboard (Source. <https://aws.amazon.com/>)

These checks can save you money by automatically:

- Identifying unused resources
- Identifying idle resources
- Recommending Savings Plan purchases
- Checking for over- or under- used instances
- Checking reservation contract expirations

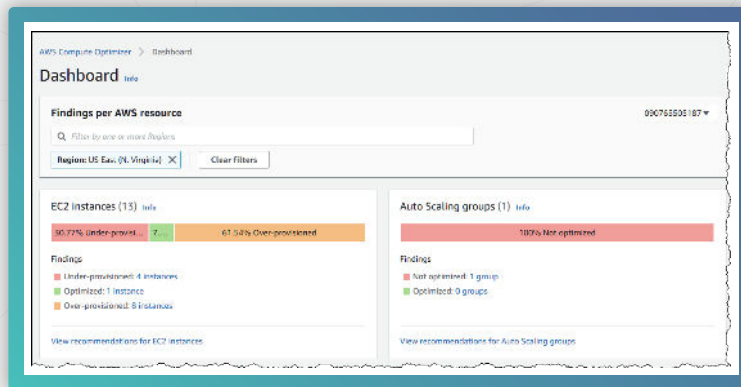
Trusted Advisor also supports email notifications that trigger when a check's status changes. You can use ongoing real-time checks and email notifications as a great way to trigger automating your scripted AWS sizing tasks.

This service offers two basic checks for free. For full access to all of Trust Adviser's checks, you must have the business or enterprise AWS support tier.

2. Sizing by Historical Workloads

The **Compute Optimizer** is a tool that provides AWS sizing recommendations for your inventory of Amazon EC2 instances. This tool analyzes historical resource utilization data (via CloudWatch) to create optimal resource recommendations for maintaining similar workloads. These recommendations commonly consist of switching instance sizes or instance types, but can also include Auto Scaling Group (ASG) and EBS volume recommendations.

The Compute Optimizer's dashboard provides a visual representation of your total under-provisioned, over-provisioned, and optimized resources so that you can quickly gauge the overall efficiency of your inventory by each resource type.



AWS Compute Optimizer Dashboard (Source. <https://aws.amazon.com/>)

When reviewing a recommendation, you can check the forecasted CPU and Memory utilization for that recommended instance type. This is great for checking if a recommendation's resource utilization is at or below an acceptable threshold (e.g., your risk tolerance for spikes in CPU utilization of a fully utilized instance).

You can save money with the AWS Compute Optimizer by analyzing any of the following workload types:

- CPU-intensive workloads
- Network-intensive workloads
- IO-intensive workloads
- Memory-intensive workloads

To get sizing recommendations based on memory utilization and other operating-system level metrics, you must install a CloudWatch agent to your EC2s.

For those of you who prefer using the AWS CLI, you can review recommendations as JSON. Here is an example of what that would look like:

```
$ aws compute-optimizer get-ec2-instance-recommendations --instance-arns
arn:aws:ec2:us-east-1:012345678912:instance/i-0218a45abd8b53658
{
  "instanceRecommendations": [
    [
      "instanceArn": "arn:aws:ec2:us-east-1:012345678912:instance/i-0218a45abd8b53658",
      "accountId": "012345678912",
      "currentInstanceType": "m5.xlarge",
      "finding": "OVER_PROVISIONED",
      "utilizationMetrics": [
```

```
{
  "name": "CPU",
  "statistic": "MAXIMUM",
  "value": 2.0
},
"lookBackPeriodInDays": 14.0,
"recommendationOptions": [
  {
    "instanceType": "r5.large",
    "projectedUtilizationMetrics": [
      {
        "name": "CPU",
        "statistic": "MAXIMUM",
        "value": 3.2
      }
    ],
    "performanceRisk": 1.0,
    "rank": 1
  },
  {
    "instanceType": "t3.xlarge",
    "projectedUtilizationMetrics": [
      {
        "name": "CPU",
        "statistic": "MAXIMUM",
        "value": 2.0
      }
    ],
    "performanceRisk": 3.0,
    "rank": 2
  },
  {
    "instanceType": "m5.xlarge",
    "projectedUtilizationMetrics": [
      {
        "name": "CPU",
        "statistic": "MAXIMUM",
        "value": 2.0
      }
    ],
    "performanceRisk": 1.0,
    "rank": 3
  }
],
"recommendationSources": [
  {
    "recommendationSourceArn": "arn:aws:ec2:us-east-1:012345678912:instance/i-0218a45abd8b53658",
    "recommendationSourceType": "Ec2Instance"
  }
]
```

```

}
],
"lastRefreshTimestamp": 1575006953.102
}
],
"errors": []
}

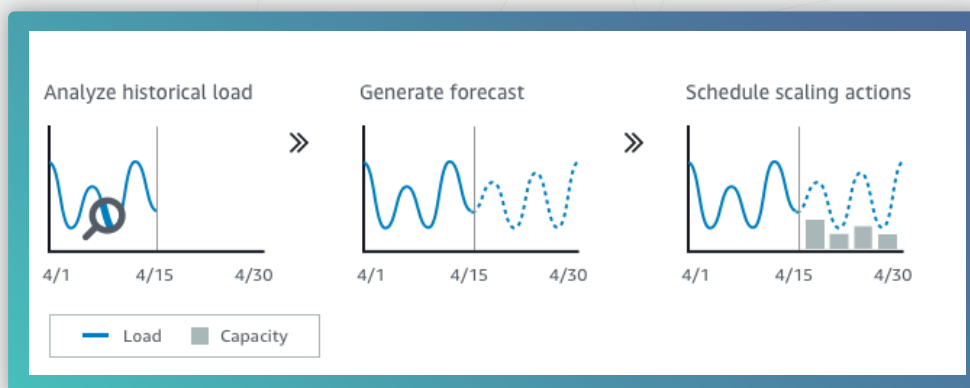
```

Overall, the AWS Compute Optimizer teaches users to be more aware of over-provisioning and under-provisioning their resources, and conduct what-if analysis. This service is available for free. To get started, simply opt into the service from within your AWS Compute Optimizer Console.

3. Sizing by Future Workload Estimations

The EC2 Predictive Scaling is a sizing feature of ASGs that allows you to scale your fleet using anticipated workload requirements. Predictive scaling works by measuring daily and weekly traffic behavior to determine when (on average) usage is expected to change, by how much, and for how long.

Predictive Scaling policies require at least one day of historical data to begin making future predictions. After setup, the policy's model evaluates activity every 24 hours to forecast EC2 usages for the following 48 hours. It then schedules these changes as scaling actions, with the goal of maintaining utilization at the target level specified in the scaling strategy.



EC2 Predictive Scaling Feature of ASG (Source. <https://aws.amazon.com/>)

You can use predictive scaling in conjunction with dynamic scaling strategies. This means that you can provision the node count in your ASG based on predicted requirements according to historical weekly and daily patterns, however for additional confidence, you can also account for the changing values of real-time metrics (such as CPU) to adjust your node count to accommodate an unexpected jump in workload. Using a predictive scaling model can trim down time spent running larger instances. And since EC2s are metered per-second and rollup into instance-hours, you can significantly lower your bill through this one AWS sizing technique alone.

4. Sizing by Data Access Patterns

S3 Intelligent-Tiering is a class of S3 storage that automatically shuttles data between a series of frequently accessed and infrequently accessed tiers as a means of providing cost savings on storage.

Here's how each tier works:

Tier	Days Since Last Retrieval	Pricing
Frequently Access	0-30 days	Standard S3
Infrequently Access	31-90 days	Standard S3 – Infrequent Access
Archive Access	91-180 days	S3 Glacier
Deep Archive Access	> 180 days	S3 Glacier Deep Archive

The data retrieval process for S3 Intelligent-Tiering is not subject to fees, however you should expect slower retrieval times for Archive Access (3-5 hours) and Deep Archive Access (12 hours).

To enable this AWS sizing feature, simply navigate to your S3 bucket's properties and create a configuration for S3 Intelligent-Tiering. Here, you can specify which objects should be considered for tiering through a variety of filters and tag selections.

5. Sizing by Delegation

AWS Fargate is a serverless compute engine for containers that automatically launches and scales the underlying EC2 cluster to match the required workload capacity. The idea is simply to delegate the management of nodes, PODs and clusters to AWS, and only be responsible for launching containers. Keep in mind that you are still responsible for selecting a size for your container, and you would pay a bit more for the convenience, however, you no longer have to worry about provisioning nodes into a cluster.

Fargate offers a pay-as-you-go model, with pricing options such as Spot, On-Demand, and Reserved — similar to those available for Amazon EC2 instances.

6. Sizing by CPU Spikes

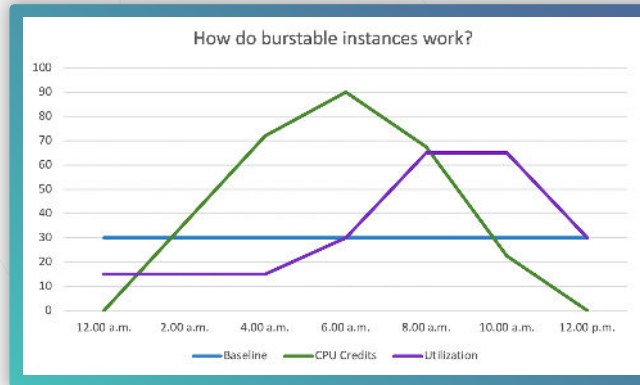
[Burstable Performance Instances](#) are instances that typically operate on a baseline-level of CPU performance, but when workload demands increase, they can temporarily burst their performance to meet demand. This burst mechanism is controlled by CPU credits that accrue hourly over time (at a rate varying by instance size) when the instance is performing at a baseline level.

T2, T3, T3a, and T4g are the only instance families that support the Burstable Performance feature. You can use Burstable Performance Instances for On-Demand, Spot, and Reserved Instances purchasing options.

Credits are consumed at the following rates:

- **100% Utilization of One vCPU:** One minute
- **50% Utilization of One vCPU:** Two minutes
- **25% Utilization of Two vCPUs:** Two minutes

Credits accrued for an instance do not expire, however, there is a limit to how many credits an instance can hold.



Burstable CPU Performance feature of T2, T3, T3a, and T4g (Source. <https://aws.amazon.com/>)

This instance type suits general-purpose applications, such as microservices, small and medium-sized databases, interactive applications with low-latency, development, virtual desktops, stage environments, build, product prototypes, and code repositories.

AWS offers a neat feature known as “unlimited” which means that there is no limit to your CPU usage, however you are charged for it in the form of surplus credits. This helps if you have an important CPU-intensive workload that is unpredictable in nature. Remember that T3a and T4g are configured as “unlimited” by default which can unintentionally result in an ever-increasing bill.

While choosing the instance size, make sure that it passes the minimum memory requirements of the underlying operating system and applications. If the application has graphical users, then the operating system consumes significant memory and CPU resources — therefore, the instance size should be large enough for many use cases. Based on the memory and CPU growth requirements over time, scaling to a larger instance type will be required.

Conclusion

Although sizing your resources can be a challenge, there are several approaches you can take to make the task easier and less risky for your application stability. The delegation of sizing may be free in some cases (such as Intelligent S3), could cost you a premium for its convenience (such as Fargate), or can result in a surprise charge (such as T4g's default unlimited burstable CPU). When used strategically, it can save you time and also money.