



eBook

# The Complete Guide for **AWS** Cost Management and Cost Allocation



# The Complete Guide for AWS Cost Management and Cost Allocation

## Contents:

|   |    |
|---|----|
| <b>Introduction</b>                                       | 01 |
| <b>AWS Essentials</b>                                     | 02 |
| <b>Key Services that Contribute to the AWS Bill</b>       | 02 |
| <b>Understanding AWS Costs: Making Sense of Your Bill</b> | 03 |
| <b>The Utility Bill Analogy</b>                           | 03 |
| <b>The AWS Parallel</b>                                   | 04 |
| <b>The Solution: Tagging and Beyond</b>                   | 04 |
| <b>Managing AWS Costs</b>                                 | 04 |
| <b>The Biggest Risk: EC2</b>                              | 04 |
| <b>Tips to Manage EC2 Cost Control</b>                    | 05 |
| • Terminate Unused EC2 Instances                          | 05 |
| • Cut Oversized Instances and Volumes                     | 06 |
| • Use Private IPs   | 06 |
| • Delete Low-Usage Amazon EBS Volumes                     | 06 |
| • Use AWS' Savings Plan                                   | 06 |
| • Utilize Reserved Instances                              | 06 |
| • Buy Reserved Instances on the AWS Marketplace           | 07 |
| • Utilize Amazon EC2 Spot Instances                       | 07 |
| • Configure Autoscaling                                   | 07 |
| • Choose Availability Zones and Regions                   | 07 |

|   |           |
|---|-----------|
| <b>AWS S3</b>   | <b>08</b> |
| <b>Tips to Manage S3 Cost control</b>                     | <b>09</b> |
| • Opt for the Right Storage Class                         | 09        |
| • Set up Billing Alarms                                   | 09        |
| • Delete Unused Objects                                   | 10        |
| • Automatically Manage Objective with Amazon S3 Lifecycle | 10        |
| • Data Transfer Costs                                     | 10        |
| • Compress your Data Before Storing it                    | 11        |
| • Partition your Data Before Querying it                  | 11        |
| <b>AWS EKS</b>  | <b>12</b> |
| <b>Tips to Manage EKS Cost Control</b>                    | <b>13</b> |
| • Rightsize Instances                                     | 13        |
| • Spot Instances  | 13        |
| • Cluster Autoscaling                                     | 13        |
| • Fargate with EKS  | 13        |
| <b>AWS-Native Cost Monitoring Tools</b>                   | <b>14</b> |
| <b>AWS Cost Explorer</b>                                  | <b>14</b> |
| <b>AWS Budgets</b>  | <b>16</b> |
| <b>AWS Cost Anomaly Detection</b>                         | <b>17</b> |
| <b>AWS Tagging</b>  | <b>18</b> |
| <b>What are Tags Used for?</b>                            | <b>18</b> |
| <b>How to Implement Tagging in AWS</b>                    | <b>19</b> |
| <b>AWS Tagging: Part of a Larger Strategy</b>             | <b>20</b> |
| <b>AWS Tagging: Best practices</b>                        | <b>21</b> |
| • Make it Obvious what a Tag's Purpose is                 | 21        |
| • Define the Rules of Usage                               | 21        |

|   |           |
|---|-----------|
| • Keep it Simple  | 22        |
| • Automate it   | 22        |
| • Review it   | 22        |
| • The Untaggable  | 22        |
| • Thinking Beyond AWS   | 23        |
| <b>Virtual Tagging</b>  | <b>23</b> |
| <b>Real-World Tagging Challenges</b>                            | <b>23</b> |
| • Poor Buy-in   | 23        |
| • Poorly-Implemented Tagging Strategy                           | 24        |
| • The Inherited AWS Environment                                 | 24        |
| • The Untaggable  | 24        |
| • The Unknowable  | 24        |
| <b>Virtual Tagging: Real-World Solutions</b>                    | <b>25</b> |
| <b>Virtual Tags: Advanced Reporting</b>                         | <b>26</b> |
| <b>You can be in Control, and you Don't have to do it Alone</b> | <b>26</b> |

# Introduction

Collectively, the three major public cloud vendors harvested 65% of the total global spend on cloud computing. In Q1 of 2022, Amazon Web Services (AWS) led the pack with a market share of 33%, Microsoft Azure with 22%, and Google Cloud Platform (GCP) with 10%. A [2021 benchmark report](#) reveals that AWS also leads GCP and Azure in terms of both latency and cost.

While it is clear that AWS' cloud service model is a go-to solution for many businesses, there is also no denying that managing cloud costs is part of the package.

Imagine receiving the call that the co-founder of the video-animation platform, Viddyoze, got from AWS: "Your card has been declined," says the customer services rep. "Really? I know there is money available; how much is the charge?" asks David. "\$65,345," says customer services. Viddyoze had anticipated paying \$10,000 that month.

Like many businesses, Viddyoze uses AWS to scale up and down with demand. In the scenario that ran up the surprise bill, something broke: as they scaled, a resource failed to deploy, so architecture was scaled and scaled again.

The good news about this story is that when David raised a case, AWS reduced the bill to around \$26,000. The bad news is that scenarios like these can cause real damage to businesses. And, it doesn't take something to break to end up paying more than you need to.

Flexera's 2022 State of the Cloud Report estimate that 32% of their respondent's cloud spend is wasted – up from 30% in the previous year. Couchbase puts those estimates even higher: with excess spend estimates by enterprises accounting for 35% of cloud costs.

Not only are businesses struggling to control their expenses; they often struggle to even understand them, let alone understand the true ROI of each customer or service.

This comprehensive eBook delves deep into the world of AWS economics, breaking down the complexities of your cloud bill and pinpointing areas of inefficiency. We'll discuss why merely glancing at top-level costs can be misleading and how these figures can mask crucial details that directly impact your bottom line. A significant portion will focus on tagging in AWS—a pivotal strategy for enhancing visibility over your cloud expenditure. Through effective tagging, you can align your AWS investment with the specific unit economics that are integral to your business's success.

# AWS Essentials

Navigating the vast realm of AWS necessitates a deep understanding of its foundational elements, particularly when it comes to deciphering the complexities of billing and managing costs.

At the heart of AWS are core services like Amazon Elastic Compute Cloud (EC2) for compute, Amazon Simple Storage Service (S3) for storage, and Amazon Elastic Kubernetes Service (EKS) for managed Kubernetes solutions- each playing a vital role in building up the overall AWS bill.

These services, while powerful and integral to diverse digital initiatives, come with their respective pricing intricacies.

## Key Services that Contribute to the AWS Bill

**Amazon Elastic Compute Cloud (EC2):** EC2 stands as one of the pillars of AWS. It provides resizable compute capacity in the cloud, allowing users to run virtual servers as per their needs. Costs here are determined by the types and number of instances you run, and can also include additional charges for added software or services.

**Amazon Simple Storage Service (S3):** S3 is AWS's scalable object storage service. It's a popular choice for backup and storage, but costs can ramp up based on the volume of data stored, retrieval rates, and any additional features like data replication across regions.

**Amazon Elastic Kubernetes Service (EKS):** A managed Kubernetes solution, EKS has seen a surge in popularity with businesses leaning into containerized applications. While EKS itself has a fixed hourly rate for cluster management, additional costs can arise from worker nodes, data transfer, and other integrated AWS services.

**Data Transfer and Bandwidth:** While often overlooked, the data going in and out of AWS services (especially to other regions or the public internet) can incur significant charges. It's crucial to monitor these data transfers to avoid unexpected surges in costs.

**Additional AWS Services:** AWS offers a wide range of other services, from databases (like Amazon RDS) to machine learning tools (like SageMaker). Each of these services has its pricing model, and even if they represent smaller slices of the bill, their collective contribution can be substantial.

AWS billing isn't just a simple invoice but a detailed view of how each service contributes to the monthly expenditure. Beyond merely reading this bill, businesses need to master cost management strategies to ensure efficient resource utilization.

By diving into these aspects, and continually optimizing usage against costs, businesses can ensure that their AWS journey is both powerful and cost-effective.

## Understanding AWS Costs: Making Sense of Your Bill

As organizations increasingly rely on AWS to secure their digital infrastructure, one recurring challenge stands out: understanding the AWS billing structure due to the disconnect between service usage and the associated costs. The bill, though comprehensive, often presents complexities that make it challenging to understand and anticipate exact costs and allocations. This chapter aims to explore this gap, drawing parallels with everyday expenses, and introducing strategies to gain insights into the Unit of Economics pivotal to your business.

### The Utility Bill Analogy

Let's take a moment to relate AWS costs to something more familiar: our monthly utility bills.

Imagine receiving your electricity bill. It clearly indicates the units of electricity consumed over the month. While this information is crucial, it lacks granularity. You know your overall consumption but remain in the dark about the individual energy costs of, say, heating your office or the excess used in heating an empty room.

Similarly, consider your broadband expenses. You're billed a set amount monthly, reflecting your total internet usage. But, bundling this with your electricity doesn't provide a clear picture of the individual costs of keeping everyone at home warm and connected. Does one person's online activity consume more resources than another's?

## The AWS Parallel

Much like the utility scenario, in AWS, the resources you use don't directly translate to the services you deploy. Knowing you've used X number of EC2 instances or Y amount of S3 storage is just the tip of the iceberg. The challenge is understanding which business operations or specific customers are driving this consumption, and evaluating the return on investment they provide.

## The Solution: Tagging and Beyond

The initial solution might seem simple: tag every resource based on specific criteria, like projects or owners. This can be invaluable for a majority of AWS services. But, complications arise when you incorporate dynamic orchestrators, such as Kubernetes. With the vast number of resources it deploys, tagging every single Kubernetes entity within the cluster becomes an overwhelming, if not impossible, task.

However, from a broader business viewpoint, the essence remains: you need a clear understanding of your cloud consumption to effectively calculate and manage unit costs.

As we dive deeper into this guide, we'll reveal methods and best practices to achieve this clarity, ensuring that you're not just aware of what you're spending, but also achieve maximum value from every dollar invested.

## Managing AWS Costs

Before diving into tagging, let's discuss the areas of cost control that do not require a large investment to implement, and which also cut to the heart of the matter – controlling the services that pose the greatest risk to your budget.

### The Biggest Risk: EC2

AWS' computing products and services offering include servers, networking, storage, remote computing, mobile development, email, and security. And the riskiest of those, in terms of cost management? AWS' EC2.

Elastic Cloud Compute (EC2) is a cloud platform that offers secure and resizable computing capacity. Instances are the building blocks of computing capabilities, and Amazon EC2 provides 5 instance types to suit various use cases:



**General-purpose instances:** These offer balanced memory, computing, and networking resources and are suitable for gaming servers, application servers, and back-end servers for companies.

**Compute-optimized instances:** These are excellent for applications requiring high computing power. Examples of such use cases include scientific modeling, dedicated gaming servers, and machine learning.

**Memory-optimized instances:** These are great for the quick delivery of workloads involving large datasets, particularly when enormous amounts of data need to be preloaded before running an app.

**Accelerated computing instances:** These use hardware accelerators that boost data processing, making them best suited to running graphics applications and streaming.

**Storage-optimized instances:** These are suitable when there are large datasets on local storage, such as at data warehouses and online transaction systems.

Cloud architecture must be responsive to changing business needs, but to accommodate this, cloud teams tend to over-engineer to create reliable margins for error. However, the end result of this caution is “bad sprawl”, which must be controlled. This can be a real challenge, especially in EC2, when you use the elastic, i.e., scalable, aspect of the service. Budgets are relatively simple if you sign up for a dedicated instance. However, as soon as you add scale to your application, you need to keep a tighter rein on your budgets. With this in mind, let’s take a quick look at some low-investment strategies companies can implement to immediately improve how they manage their AWS costs.

## Tips to Manage EC2 Cost Control

In the next chapter, we’ll guide you through practical steps to manage and control these costs effectively. Think of it as your roadmap to making the most of EC2 without breaking the bank. Let’s dive in and explore how to use EC2 smartly and cost-effectively.

### Terminate Unused EC2 Instances

Using AWS Cost Explorer Resource Optimization, you can get a report of idle or low-utilization instances. Once you identify these instances, you can stop or downsize them. Once you stop an instance, you must also terminate it. This is because if you stop an

instance, your EBS costs will still be incurred. By terminating EC2 instances, you will also stop EBS and EC2 expenses.

## Cut Oversized Instances and Volumes

Before deciding which instances and volumes need to be reduced, an in-depth analysis of all available data is required. Do not rely on data from a short period of time. The time frame for a data set should be at least one month, and make sure to take into account seasonal peaks. Remember that you will not be able to reduce Elastic Block Storage (EBS) volumes. So, once you know the appropriate size you require, create a new volume, and copy the data from the old volume.

## Use Private IPs

Whenever you communicate in the Amazon EC2 network using public IPs or an Elastic load balancer, you will always pay Intra-Region Data Transfer rates. Use private IPs to avoid paying this extra fee.

## Delete Low-Usage Amazon EBS Volumes

Track EBS volumes for at least 1 week and identify those that have low activity (at least 1 input/output per second per day). Take a snapshot of these volumes (in case you will need them at a future date) and then delete them.

## Use AWS' Savings Plan

AWS' Savings plan is a flexible pricing model running for one to three years. In this model, you pay a lower price on EC2 and Fargate usage for a promise of a steady amount of usage during the specified period. The agreed usage amount is usually discounted by more than 30%. The AWS Savings Plan is ideal for stable businesses that know their resource requirements.

## Utilize Reserved Instances

By reserving an instance, you may save up to 70%. But, if you don't use the reserved instance as much as you expected, you may end up overpaying instead. This is because you will pay for 24/7 utilization for the entire reserved period, regardless of whether you used the resource or not.

## Buy Reserved Instances on the AWS Marketplace

The AWS Marketplace is like a stock market. You can sometimes buy Standard Reserved Instances at extremely affordable prices in comparison to buying directly from AWS. In this way, you can end up saving almost 75%.

## Utilize Amazon EC2 Spot Instances

Spot instances can reduce costs by almost 90%. Spot instances are great for workloads that are fault-tolerant, for example, big data, web servers, containerized workloads, and high-performance computing (HPC). Auto-scaling automatically requests spot instances to meet target capacity during interruptions.

## Configure Autoscaling

Autoscaling allows your EC2 fleet to increase or shrink based on demand. By configuring autoscaling, you can start and stop instances that don't get used frequently. You can review your scaling activity using the CLI command. Review the results to see whether instances can be added less aggressively or to see if the minimum can be reduced to serve requests with smaller fleet sizes.

## Choose Availability Zones and Regions

The cost of AWS varies by region. Data transfers between different availability zones are charged an extra fee. It is, therefore, important to centralize operations and use single availability zones whenever possible.

Effectively managing EC2 costs is paramount for organizations aiming to harness the power of AWS without incurring unnecessary expenses. EC2, with its vast array of instance types and configurations, can be both an asset and a challenge. As highlighted, potential pitfalls, such as unused or oversized instances, can quickly inflate costs if not managed properly.

## AWS S3

Amazon Simple Storage Service (Amazon S3) is one of the most widely used cloud storage services, offering scalability, performance, durability, availability, and scalability. While it's known for its cost-effectiveness, managing its costs can be challenging, especially when usage scales.

What does S3 include?

Amazon S3 provides object storage, which means it stores data as individual objects within buckets. Each object offers distinct advantages based on the specific requirements of data storage, access frequency, and budgetary considerations:

**Standard:** A general-purpose storage class ideal for frequently accessed data. It offers high durability and availability, making it suitable for big data, content distribution, and backups.

**Intelligent tiering:** Designed for data with unpredictable access patterns. This class automatically shifts data between frequent and infrequent access tiers to optimize costs without compromising performance.

**Infrequent access (standard-IA):** This is a cost-effective solution for data that's less frequently accessed but requires rapid access when needed, like long-term backups or older data that are accessed sporadically.

**One-zone infrequent access:** A more affordable option that stores data in a single availability zone. It's ideal for infrequently accessed data that can be recreated if lost, such as secondary backups or temporary storage.

**Glacier:** An archival storage solution for data that's seldom accessed and can tolerate retrieval times ranging from minutes to several hours. It's best for long-term backups and archives with extended retention periods.

**Glacier deep archive:** The most cost-efficient archival class, perfect for data that might only be accessed once or twice a year. It's primarily used for long-term retention of data that must be preserved for years, often fulfilling regulatory requirements.

Now that we are familiar with AWS S3 objects, let's check out the factors impacting S3 pricing.

The main components for S3 pricing include:

**Storage costs:** Charges are based on the amount of data stored in S3, typically priced per GB/month.

**Request costs:** Charges for the number and type of requests (such as GET, PUT, and DELETE). Different types of requests have different costs.

**Data transfer costs:** Charges for data going out of S3 to the internet or to other AWS regions. Transferring data into S3 is generally free.

**Additional feature costs:** S3 offers various features like versioning, object tagging, data transfer acceleration, and more. Some of these features come with additional costs.

## Tips to Manage S3 Cost Control

Opt for the Right Storage Class

**Standard:** For frequently accessed data.

**Intelligent-tiering:** For data with changing access patterns.

**One zone-infrequent access:** For infrequently accessed data stored in a single zone at a lower cost.

**Glacier and glacier deep archive:** For long-term archival at much lower costs, albeit with slower retrieval times

Set Up Billing Alarms

Once you have the right storage class for your use case figured out, considering the cost of S3 storage, the next step is creating a billing alarm to avoid a surprisingly high S3 bill.

You can do that through the CloudWatch console by choosing 'Billing' and 'Total Estimated Charge' as your metric, effectively monitoring your Amazon S3 costs. From there, you can define a dollar value that triggers your alarm and sends a notification when your usage exceeds that threshold.

## Delete Unused Objects

In S3, each object version contributes to your storage AWS S3 costs. But how do you find unused objects within a bucket?

One way to reduce S3 costs is by using the AWS CLI to list all objects in a specified S3 bucket. Simply run `aws s3 ls --summarize --human-readable --recursive s3://your-bucket-name`.

Then, identify the unused objects by looking at the LastModified column. If empty, delete the object by running `aws s3 rm s3://your-bucket-name/object-name`.

You can also delete multiple (or all) objects within a bucket as well as entire buckets. Just keep in mind that S3 won't be able to restore any data after the fact.

## Automatically Manage Objects with Amazon S3 Lifecycle

With Amazon S3 Lifecycle, you can automate the transition of objects to less expensive storage classes at set intervals and specify when objects should be deleted from your Amazon S3 bucket.

For example, if you know that certain objects are infrequently accessed, you might want to transition them to the S3 Standard-IA storage class. Or, you can set up S3 to automatically delete expired objects on your behalf.

## Data Transfer Costs

Minimize data egress by strategically placing resources. You can use the [cross-region replication feature](#) to mirror your S3 bucket in a different region.

For example, if you are transferring 20 GB from a bucket in US-west-2 to an EC2 instance in US-east-1, you would be charged \$0.20. However, if you first downloaded the data to a mirror S3 bucket in US-east-1, you would only pay \$0.02 for transfer and \$0.03 for storage over the course of a month, which would be significantly less expensive.

## Compress your Data Before Storing it

S3 charges you based on the amount of data you store and the amount of data you transfer out of the service.

By compressing your data before you send it to S3, you can reduce the amount of storage and data transfer charges you pay. Your best bet is to use fast compression such as LZ4 to improve performance while reducing storage requirements and AWS S3 storage costs.

## Partition your Data Before Querying it

The number of requests used to access your data as well as their type will affect your AWS S3 costs.

To minimize them, consider partitioning your data. Partitioning improves query performance because it limits the amount of data that has to be scanned. For example, if a table is partitioned by date, a query that accesses only a single day of data can scan only the files in that day's partition, rather than the entire table.

Navigating the intricate landscape of AWS S3 costs requires a blend of understanding its service offerings and deploying best practices. As we've delved into the myriad storage classes, from frequently accessed data types to deep archival solutions, it's evident that S3's versatility offers both an opportunity for optimized cost savings and the potential for oversights leading to inadvertent expenses. Implementing strategies, whether it's selecting the right storage class, setting up billing alerts, or utilizing S3 Lifecycle for automation, helps in curbing unexpected charges. Additional steps, such as compressing data, partitioning, and strategically placing resources, further enhance cost efficiency.

As we wrap up this section, it's important to remember that proactive management, continuous monitoring, and informed decisions are the keys to harnessing the power of S3 without straining your budget. Whether you're a new or a seasoned AWS user, these insights are invaluable in ensuring a cost-effective S3 experience.

# AWS EKS

Amazon Elastic Kubernetes Service (EKS) is a managed service that allows users to run Kubernetes on AWS without needing to install, operate, or maintain their own Kubernetes control plane. As businesses scale and deploy more containerized applications, the costs associated with EKS can grow. Understanding the nuances of EKS billing and identifying opportunities for cost optimization becomes paramount.

EKS is priced at \$0.10 per hour for each cluster you create – you can use a single cluster to run multiple applications using Kubernetes namespaces and IAM security policies. If you use EKS with EC2, you'll be charged for the resources you created to run your Kubernetes worker nodes. That is the on-demand price and you will pay for what you use.

If you choose to use EKS Fargate, you don't need to pay for provisioning and managing servers costs but pricing is based on the CPU and memory resources used from the time you start to download your container image until the Amazon EKS pod terminates (minimum 1-minute charge).

Also, if you want to use on-premise EKS, you can use EKS on AWS Outposts. Amazon EKS on AWS Outposts pricing is simple and works the same as it does in the cloud: you pay \$0.10 per hour per cluster for the Amazon EKS service. This applies for both extended and local cluster deployment options. The Amazon EKS service fee is not included in the outposts pricing.

When you deploy Amazon EKS, you are billed for:

**EKS control plane:** Amazon EKS has a set price per hour for each cluster that you run, regardless of its size or configuration.

**Worker nodes:** These are Amazon EC2 instances, and you'll be billed for them based on the type and number of instances you deploy.

**Data transfer:** Charges apply for data moving out of EKS to the internet or other AWS services.



**Additional AWS services:** Depending on your architecture, you might also utilize and be billed for other AWS services like Amazon RDS, ELB, or EBS volumes.

## Tips to Manage EKS Cost Control

One of the main reasons why teams see their EKS bills surge is due to over provisioning. You can avoid this by choosing the best instance types, implementing cost monitoring and alerting strategies, and promoting a culture of financial accountability.

### Rightsize Instances

Regularly monitor the CPU and memory usage of your nodes. If they are consistently underutilized, consider switching to a smaller EC2 instance type..

### Spot Instances

You can use Spot Instances to save up to 90% off pay-per-use pricing models. The only problem is that spot Instances aren't suitable for fault-tolerant workloads because of potential service disruptions. You can use Spot Instances with Batch Processing Tasks, CI/CD Servers, High-Performance Computing, Big Data Analytics, and Rendering Workloads while using EKS and you can reduce the EKS costs.

### Cluster Autoscaling

Cluster autoscaling is an important feature of EKS because it allows you to scale resources depending on the load on your system. If your system load decreases, the nodes will scale down automatically, preventing you from incurring unnecessary costs. To understand which resources will benefit most from autoscaling, you'll need a monitoring system.

### Fargate with EKS

AWS Fargate pricing is based on usage (pay-per-use). There are no upfront charges here as well. There is, however, a one-minute minimum charge. All charges are also rounded up to the nearest second. You will also be charged for any additional services you use, such as CloudWatch utilization charges and data transfer fees.

As opposed to Amazon Lambda, Fargate gives you the option to use different runtimes with EKS or ECS. It is also cheaper to execute Fargate than Lambda per hour. For more information, see our [AWS Fargate vs AWS Lambda comparison](#).

This chapter offered a deep dive into the range of AWS's main services, each bringing its own set of pricing intricacies and optimization opportunities. Starting with EC2, the foundation of AWS computing services, we recognized its potential for budgetary risks, especially when the elastic aspect is leveraged without adequate foresight. Strategies for managing EC2 costs, from rightsizing instances to adopting autoscaling, demand both technical expertise and financial diligence. Shifting to S3, AWS's hallmark storage solution, we acknowledged its expansive utility, from everyday data storage to deep archiving. Efficient management here pivots on understanding storage classes, being vigilant about unused objects, and leveraging features like S3 Lifecycle.

Finally, with EKS, the bridge to containerized application deployment, costs can escalate with scaling. Critical insights into rightsizing, Spot Instances, and the contrast between Fargate and Lambda illustrate the importance of being astute to avoid over provisioning and excessive bills.

Across all these platforms, the recurrent theme is clear: AWS provides robust tools for any enterprise's needs, but maximizing its potential while maintaining budgetary control demands consistent monitoring, periodic check-ins, and a proactive approach to cost management.

Having gained a solid foundation in the primary AWS services that impact monthly outlays, it's time to turn our attention to AWS's proprietary cost-monitoring tools—a crucial next step in refining our AWS cost-management strategy.

## AWS-Native Cost Monitoring Tools

AWS provides several tools to both monitor spending and assist you in controlling it:

### AWS Cost Explorer

The [AWS Cost Explorer API](#) is a native cost observability platform that provides a dashboard for users to visualize, assess, and control AWS costs over time. This dashboard allows you to enable data analysis of your cloud spend so that you can identify trends, cost drivers, and anomalies.

The main features and benefits of Cost Explorer include:

- Quick startup with pre-configured views
- Ability to set time intervals and granularity – either monthly, daily, or custom
- Filter or group data sets
- Plan up to 12 months ahead with forecasting features
- Capability to save your progress when you find a helpful view
- Build custom applications
- Get recommendations on Reserved Instances

It's important to note that Cost Explorer uses the same dataset as [AWS Cost and Usage Reports](#).



[Image from AWS blog](#)

## AWS Budgets

AWS Budgets assist when you utilize dynamic cloud usage and want to set a notification for when cost or usage meets a predefined threshold. This enables you to closely monitor Reserved Instances (RI) and Savings Plans with custom tracking. The main features and benefits of AWS Budgets include:

- Custom budgets based on your AWS environment
- Set up event-driven alerts and reports
- Receive automated daily, weekly, or monthly Budget Reports
- Create annual, quarterly, monthly, or daily budgets

Using AWS Budgets, you can create six types of budgets: Cost budgets, usage budgets, reserved instance (RI) utilization budgets, RI coverage budgets, Savings Plans utilization budgets, and Savings Plans coverage budgets.

**Create Budget**

**Budget details**

Name\* Monthly EC2 Budget

Include costs related to Service

EC2

Apply Cancel

Period Monthly

Start date\* 2015-06-01

End date\* 2015-09-30

Monthly amount\* 80.00

**Notifications**

You can create a billing alarm to receive e-mail alerts when your AWS charges exceed a threshold you choose.

Notify me when Actual costs exceed 80 % of budgeted cost

+ Add new alert

Email contacts\*

\* Required

Cancel Create

| Category                        | Cost (\$) |
|---------------------------------|-----------|
| Last six month average          | ~\$58     |
| Last full month (May)           | ~\$62     |
| Next full month forecast (July) | ~\$52     |

[Image from AWS blog](#)

## AWS Cost Anomaly Detection

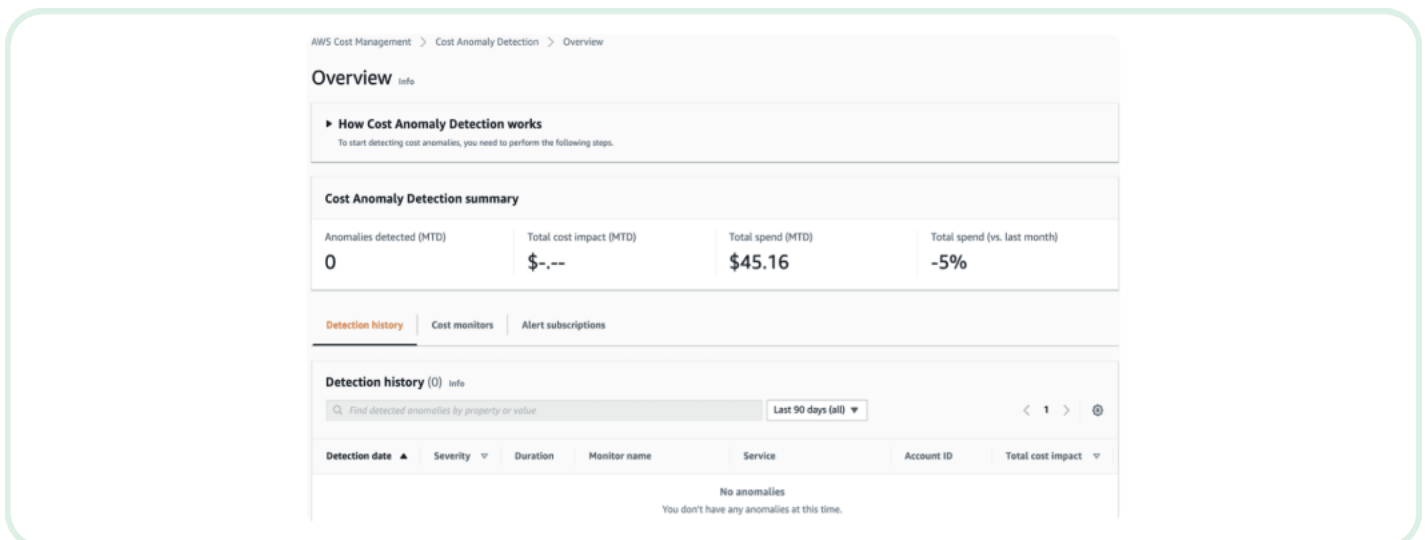
To stay informed on any irregular spending, you can set up [AWS Cost Anomaly Detection](#) via the AWS Cost Explorer API or directly in the Cost Management Console. This service creates cost-monitoring filters so that you can quickly identify and analyze anomalies. AWS Cost Anomaly Detection allows you to:

- Analyze all your AWS services independently or by member accounts, cost allocation tags, or cost categories
- Apply Machine Learning to minimize false positives
- Reduce surprise costs
- Analyze root causes that impact spending trends
- Receive individual alerts as well as daily or weekly summaries

Anomaly detection is vital because, without discovering the root cause of unusual spending, you cannot fully control your cloud spend.

Outside of the AWS Cost Management Console, you can utilize the [AWS Cost and Usage Reports](#) for comprehensive spending information or work directly with an AWS Trusted Advisor to set cost ceilings.

While this guide will take a deep dive into the more advanced methods to manage cost control than those offered by the AWS-native tooling alone, let's first consider the cost-saving strategies that can provide an immediate impact.



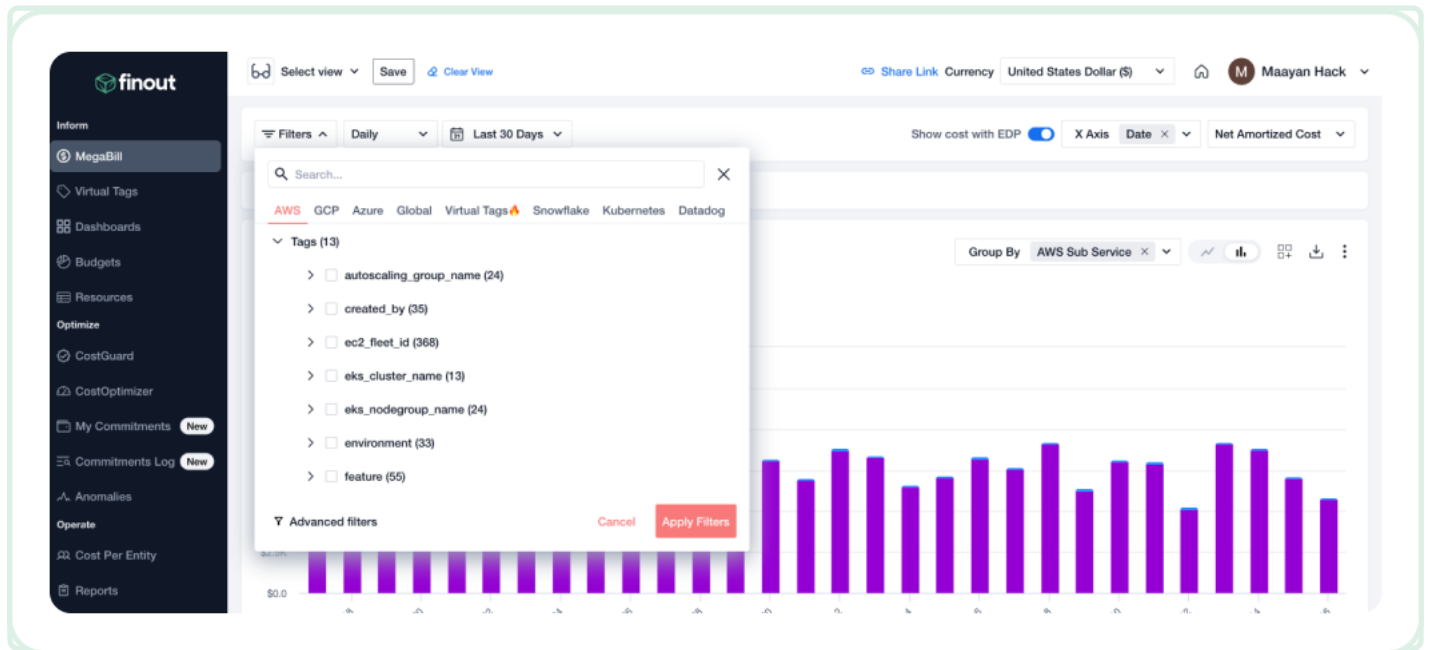
The screenshot shows the AWS Cost Anomaly Detection Overview page. At the top, there is a breadcrumb trail: AWS Cost Management > Cost Anomaly Detection > Overview. The main heading is "Overview" with an "Info" link. Below this, there is a section titled "How Cost Anomaly Detection works" with a sub-heading "To start detecting cost anomalies, you need to perform the following steps." The next section is "Cost Anomaly Detection summary" which contains a table with the following data:

| Anomalies detected (MTD) | Total cost impact (MTD) | Total spend (MTD) | Total spend (vs. last month) |
|--------------------------|-------------------------|-------------------|------------------------------|
| 0                        | \$-.-                   | \$45.16           | -5%                          |

Below the summary, there are three tabs: "Detection history" (selected), "Cost monitors", and "Alert subscriptions". The "Detection history" section shows "Detection history (0)" with an "Info" link. There is a search bar with the placeholder text "Find detected anomalies by property or value" and a dropdown menu set to "Last 90 days (all)". Below this is a table with the following columns: "Detection date", "Severity", "Duration", "Monitor name", "Service", "Account ID", and "Total cost impact". The table is currently empty, and a message at the bottom states "No anomalies. You don't have any anomalies at this time."

# AWS Tagging

AWS tags are labels that may be applied to resources such as EC2 instances and S3 buckets. They are a simple way for teams to add meaningful data to resources. AWS-native tags are added to resources by AWS; these you can't edit. Then, there are user-defined tags. A user-defined AWS tag is an assigned category or key and an (optional) value to that key, as per the traditional key:value pair.



## What are Tags Used for?

Most teams leverage user-defined tags, i.e., add metadata, for cost analyses. AWS Cost Explorer uses tags to reveal a breakdown of your AWS resource usage. And, of course, your cloud cost management platform, such as Finout, consumes these tags to perform both of these functions and much, much more.

Note that AWS tagging can also be used beyond the realm of cost control to assist with important functions such as identifying:

- Who / which team is responsible for an AWS resource?
- Which of your services have alerts enabled?
- Which of your AWS resources are unnecessary at low-load hours?
- Which roles or team members should have access to a resource?

For example, one of the most common uses of AWS tags is to use the 'Name' tag to assign a human-readable description of a resource instead of an instance ID number. For a SaaS company, this description could be the name of a customer, allowing operations teams to instantly identify which resources are being used by which clients and, by extension, how much those clients cost.

Implementing a meaningful tagging strategy is an important part of cost control in AWS, and tagging is an interdisciplinary process. That means creating a team of stakeholders from Finance, DevOps, Product, and beyond – to ensure that the tagging provides the meaningful and actionable data that the business needs.

## How to Implement Tagging in AWS

When your management tools don't give teams access to the data they need, then cloud cost control, resource monitoring, and optimization attempts are all hampered. The ultimate outcome of data collection is the information it provides, and your task as the cloud tagging team is to ensure that you don't suffer from the "garbage in, garbage out" issue. Good, clean data is essential for cloud management. To ensure you create meaningful data, pay attention to the three factors that contribute to a well-implemented AWS tagging strategy:

- Utility: tags are useful, pronounceable, and memorable

- Consistency: tags are correctly applied across all resources

- Relevance: tagging policies are reviewed to ensure that they are still fit for purpose

A strong tagging scaffold relies heavily on that last item: relevance. Relevant tagging policies are driven by teams who take the time to plan a coherent tagging strategy. The [FinOps revolution](#) has risen from the need for such teams, and implementing a coherent tagging strategy is a vital first step in ensuring their effectiveness.

Your tagging strategy begins when the stakeholders involved can provide a simple, yet persuasive, justification for each tag.

Armed with a relevant and practical tagging strategy, the next challenge is to ensure that it is applied consistently. This means that each tag must be assigned to every

resource that applies. Note, this does not equate to “create hundreds of tags”; it means that the correct tags are applied to the relevant resource.

And finally, there is utility. For those on the tagging strategy team, please be considerate of those who will use your tags to extract information. A tag that's easy to pronounce and remember is far more effective than a random or unreadable phrase. Intuitive tags lead to consistent application and simplified reporting, benefiting everyone involved.

And so, we cycle back around to that last item again: relevance. Creating a cloud tagging strategy is not a one-off exercise. Sure, that task should never be so large as the first time, but fine-tuning your tags in response to evolving business needs is part of maintaining their utility and relevance.

## AWS Tagging: Part of a Larger Strategy

Tagging, in and of itself, will not control costs or manage resources. A tagging strategy is simply the data layer that supports your resource management. It must, therefore, be part of a larger strategy. This strategy will be supported by at least three layers:

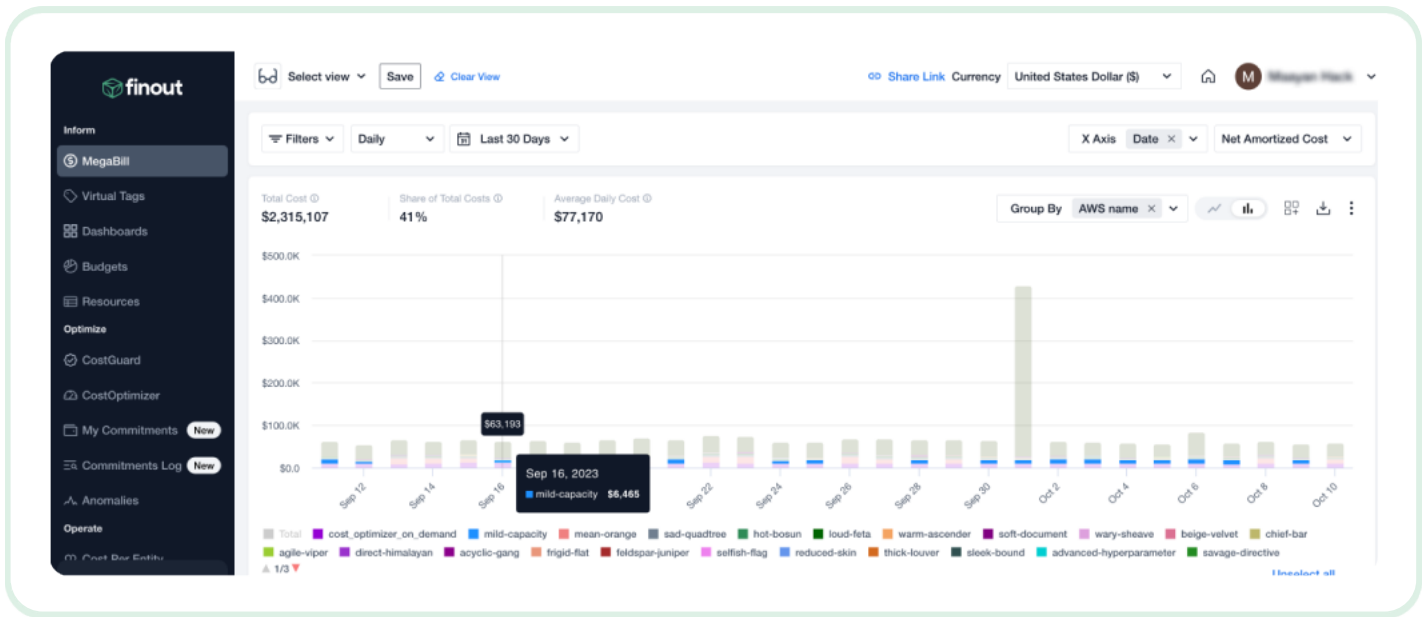
**Control layer:** There must be an approval process for approving new cloud builds and a review process to identify opportunities to decommission underutilized or inactive resources. This will be so much easier now that you have tags that identify the owner or team responsible for each resource!

**Governance layer:** There must be a clearly defined schedule to specify how often cloud resource usage must be reviewed. The team must also establish how system resources should be evaluated in terms of the value they provide.

**Reporting layer:** The reporting layer consumes the data provided by tagging and other cloud monitoring solutions. Such reporting does not have to be restricted to describing the status quo; once the data pipeline is in place, it is possible to schedule routine forecasts of anticipated cloud spend projections. Comparing these to the actual cloud spend can reveal those unanticipated issues.



# AWS Tagging: Best Practices



You can apply the following proven strategies to improve your AWS tagging policy:

## Make it Obvious What a Tag's Purpose is

AWS recommends creating four categories of tags: Technical, business, security, and automation. You may have your own favorite pillars. But, whatever you select, make sure the strategy is obvious to new users.

Beware of those inherited tags! It is not uncommon for teams to be fairly experimental in their approach when they first dip their toes in the cloud. The issue is that such resources may remain active in your cloud environment, and no one wants to terminate them, as they are unsure what they were set up for.

## Define the Rules of Usage

Note that the value component of the key:value pair is optional in AWS, but is it optional in your ruleset? It must be clear which tags are mandatory and which are optional. AWS does have naming restrictions, so align your AWS tag naming convention with this. Establish whether "-" or "\_" is allowed, or whether you will use PascalCase or camelCase.

Variants can cause big problems. Make sure that everyone understands that AWS tags are case-sensitive: ClientA, clienta, and clientA are three different entities in AWS. The abbreviation environment to env creates a unique tag.

## Keep it Simple

AWS limits you to 50 tags per resource; try and limit yourself further. Remember what it's like to be the new brain on the team and cater to the lowest common denominator. But, in balance to that, if in doubt, tag!

Mitigate against poor buy-in: after those baby steps and those first oblique AWS bills, the need for a tagging strategy becomes apparent, and a team is given responsibility for setting it up. However, everyone is busy, thank you very much, so even if the tagging strategy is developed, it may not be enforced coherently – leaving many costs unassigned. Keep it simple to encourage uptake.

## Automate it

Amazon provides a resource tagging API. This enables you to govern and implement tags at scale. More automation means less human error: providing you with higher quality and more maintainable tags.

Automation does not mean perfection: always consider human nature. Say you are cleaning up historical tags, and the tagging cleanup team was not game for laboriously manually re-tagging those resources, so they do what devs do and assign tags using a script. This means that the outcome is only as good as the assumptions that were made to perform the assumptions (which may not always be perfect).

## Review it

Implementing your initial tagging strategy is only half of the challenge. Keeping tags relevant and updated is the other half. You will probably find that high-level concepts such as 'purpose' and 'cost center' are consistent. However, tags related to specific technologies or projects will likely have a much higher turnover; hence you should regularly review your tagging strategy.

## The Untaggable

However good your tagging strategy is, there may be resources that you simply can't tag within AWS. For example, as mentioned before, if you use a dynamic orchestrator like Kubernetes, it is not feasible to tag every single Kubernetes resource in the cluster.

This is why, even with a robust tagging strategy, FinOps teams must abstract above the cost-control level provided by AWS and, ultimately, move beyond AWS.

## Thinking Beyond AWS

Allocating the cost of the items on your cloud bill can be a highly complex task, especially if you're doing it manually, in spreadsheets, or with AWS Cost and Usage Reports. Also, if you are one of those [73% of enterprises currently use two public clouds, or 26% that use three or more](#), cost allocation requires consolidating all the available data from your cloud provider(s) into one mega bill and going beyond AWS-native tooling.

In the same VMware study referenced above, respondents from organizations currently leveraging multi-cloud strategies reported:

41% lower costs and fewer hours spent on IT infrastructure and security incidents

35% savings in productivity across a distributed workforce

There are clearly benefits to taking a multi-cloud approach. But, in balance to that, 23% of respondents in the same survey reported difficulties in optimizing spend.

So, what is the strategy to abstract to a level above the cloud bill and apply a strategy that will reveal your business' Unit of Economics? Welcome to the world of Virtual Tagging.

## Virtual Tagging

Virtual tagging solves the issues that your cloud-native tagging strategies can't.

### Real-World Tagging Challenges

There are many reasons why tagging may not be as efficient or effective as it should be:

#### Poor Buy-In

Without a solid [FinOps philosophy](#) connecting teams and empowering everyone to understand their place in cost control, it can be challenging to initiate the required level of buy-in to ensure that tagging is applied consistently.

## Poorly Implemented Tagging Strategy

If a closed list is not enforced, then tag variants are an issue. Your team may be consistent with:

Environment:Production / Environment:Staging / Environment:Development

And, maybe most of the team are consistent with:

Project:prod\_a / Project:prod\_b

But, all it takes is one person applying Project:product\_a, and your cost management strategy suddenly falls apart.

## The Inherited AWS Environment

Inconsistent tags need not arise because someone made a typo or wrote quicker than they thought. Variants in tagging approaches can arise due to employee churn or to mergers and acquisitions. It's hard enough managing one company's tagging policy, let alone combining two entirely separate sets.

## The Untaggable

Tagging teams almost certainly come up against the problem that certain resources in AWS just don't align with the desired tagging policy. It might be that the costs associated with such resources need to be abstracted up a level or two.

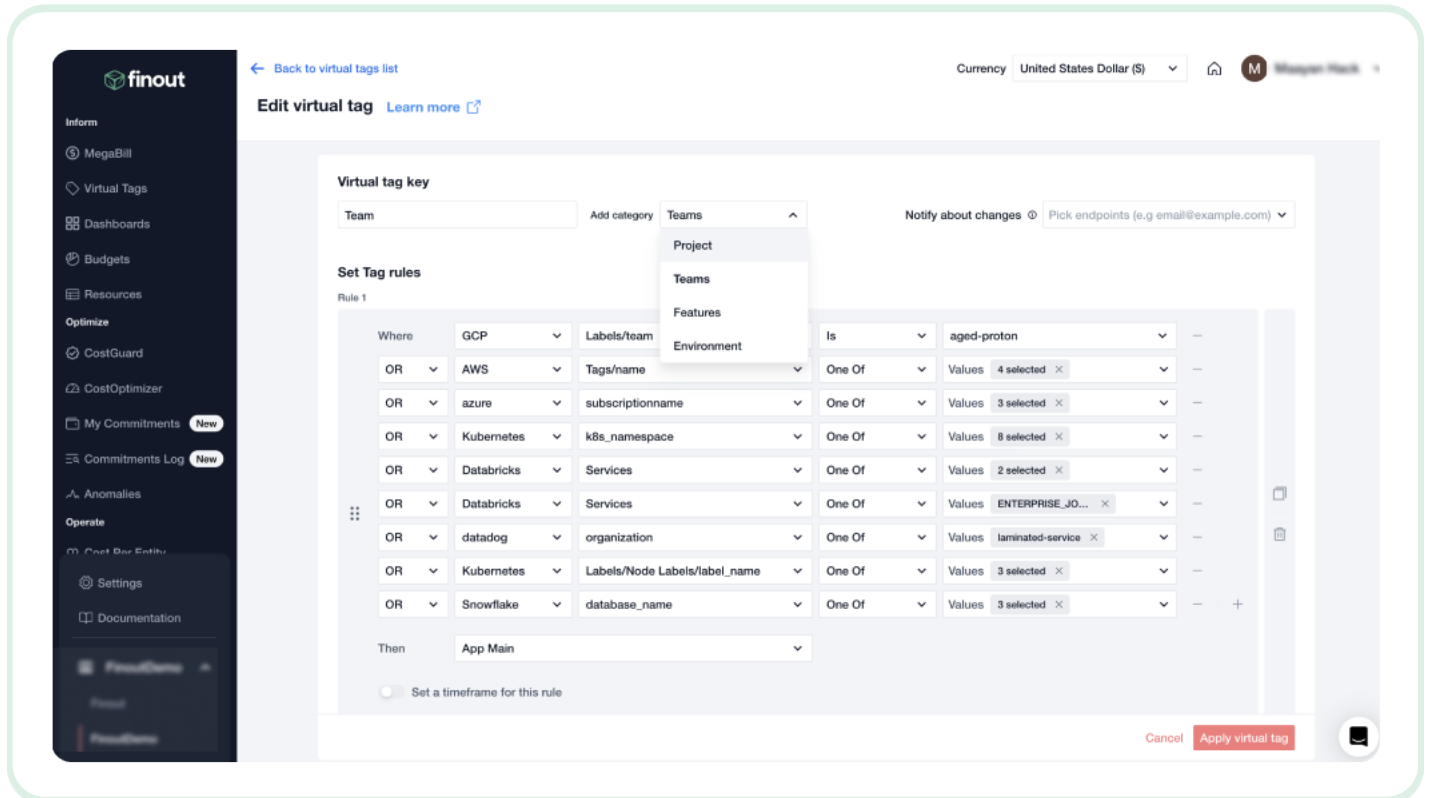
This could be due to issues such as those already raised that arise from the complexity of a Kubernetes environment.

## The Unknowable

It can also be that no one considered the abstraction required when the resources were put in place. For example, consider an eCommerce service that scaled way beyond original expectations; it might be necessary to implement a cost allocation tag as general as "DAM-infrastructure". This could help collate global costs that arise from implementing the Digital Asset Management solution so that FinOps can compare those costs to an independent supplier's IaaS offering.

## Virtual Tagging: Real-World Solutions

Virtual tags allow you to abstract away these challenges, allowing you to collate the data you need. Poorly implemented or updated tagging policies can be accommodated by adding a virtual tag abstraction layer. Virtual tagging enables FinOps to be agile and respond to those new use cases and apply logic over a different tag. There is no such thing as an “untaggable” resource at this abstraction layer. And, of course, you can collate data from a multi-cloud and enforce cost observability across all resources.



If your existing tag set has not been well controlled due to poor buy-in or people applying variants of tags, or your team has undergone a merger with another, the “Contains” query allows you to add all possible variants of a tag to generate a new collative tag.

If FinOps decides they want to run a procurement initiative and compare in-house costs with an outsourced option, they can quickly adopt a tagging strategy to collate all the relevant costs and instantly have the cost data they need.

And that Kubernetes abstraction issue, dealt with! As Figure 1 shows, labels and deployments can now be sliced and diced according to your reporting requirements.

## Virtual Tags: Advanced Reporting

Thanks to Finout's cloud cost observability platform, engineering group managers can now understand the current state of their group budget and answer vital questions, such as:

Consider "Bank INC", whose cloud provider is AWS. They use K8s heavily (like so many modern companies), and recently they added Snowflake as a scalable solution for their Big Data needs. All major R&D groups rely on the infrastructure, creating multi-tenancy cost chaos. Each month 3 different bills arrive, each one calculated and analyzed differently, while all the Head of Engineering wants to know is: "How much did each group in engineering spend in the cloud in the past 3 months?"

How much are we spending on each cloud service?

What does customer X cost us?

What has changed since last week?

What is the cost increase driver?

## You can be in Control, and you Don't have to do it Alone

Navigating the complexities of AWS cost allocation can be a challenging task, but Finout has carefully developed a solution that empowers organizations to streamline their cost management processes with precision and ease. As a holistic platform, Finout consolidates all cloud costs, providing a single source of truth for AWS, GCP, Kubernetes, Datadog, and Snowflake expenses.

The intuitive interface of Finout allows users to understand their expenditures in detail, offering the granular data essential for making informed decisions about AWS usage. If

the intricacies of AWS tagging have left you seeking alternatives, Finout's innovative virtual tags feature is the perfect remedy. This feature permits the remapping of cost filters into tag values, thereby enabling precise tracking of spending across various cloud platforms and billing origins.

A unique aspect of Finout's virtual tags is their capacity to amalgamate AWS tags, Kubernetes labels, and GCP namespaces into a unified dashboard. This simplifies the process of monitoring expenses by development teams, features, and environments, while also resolving common issues such as incorrect tagging of AWS services or inconsistent tag naming.

Moreover, Finout sets itself apart from other cloud cost monitoring tools by offering customizable dashboards, reports, and notifications tailored to meet the specific needs of your infrastructure. This functionality ensures that IT teams are proactively informed and aware of any unusual spending patterns or anomalies. The depth of insight provided by Finout allows for a comprehensive analysis of cloud usage and actual cost per customer, feature, or team. This level of visibility is instrumental in facilitating collaborative cost-cutting measures and informed decision-making.

AWS cost allocation tags are undeniably vital for organizations leveraging AWS for their cloud infrastructure. However, the manual management and implementation of these tags can often result in inaccuracies in cloud cost reports. This is where Finout shines, offering a robust suite of features including multi-cloud cost optimization, customizable dashboards, historical reports, and comprehensive cloud cost optimization. Want to have a try? [Get started today!](#)