Whitepaper

AWS Cost Optimization in Five Perspectives

A Pragmatic and Universal Cloud Cost Optimization Framework

By Michel Zitman, Mudit Gupta, Imre Blijleven and Jeroen Jacobs

Foreword.

If you leave your home, it's common sense to lower the room temperature and turn off the lights. This directly impacts your consumption of gas and electricity and thus your energy invoice. This common practice is not consistently applied in organizations around cloud spend. Cloud cost management appears to be a challenging subject in most organizations. From our experience we have concluded that 35% of cloud spend is wasted because of over-provisioned resources alone. Research from 451 Research (Nov. 2022) made it clear that only 62% of the organizations are making use of reserved instances or Savings Plans and according to the Flexera 2022 State of the Cloud Report, 32% cloud spend is estimated to be wasted. And given the fact that, on a global scale, we are only at the beginning of cloud adoption, it's time for action. Fortunately, most organizations, especially enterprises, realize that cloud cost management is a top priority.

With this whitepaper, we hope to inspire you to actively engage with cloud optimization best practices to make your cloud environment even more profitable.

"Are you in control of your cloud spend?"

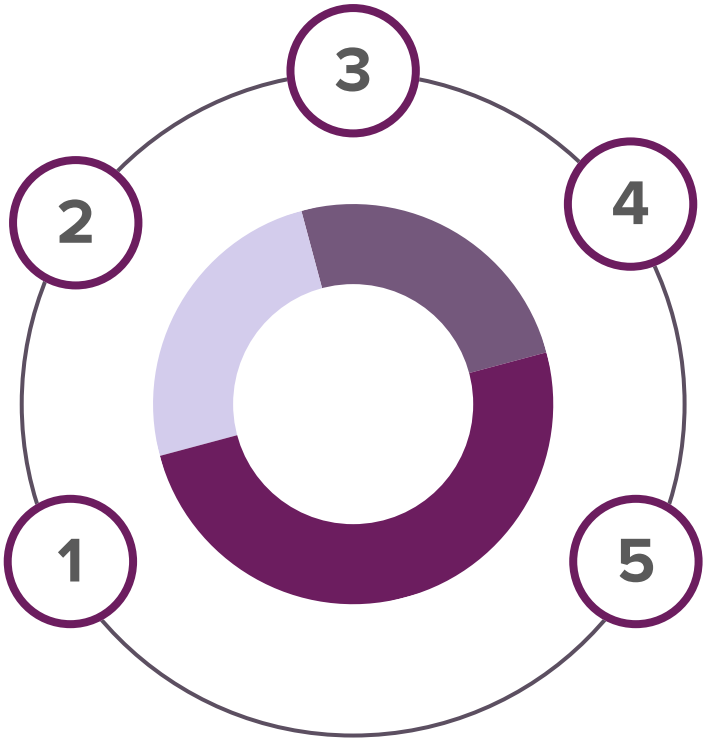
Index

— Introduction

— Perspectives and Conclusions

Automate your controls.

The Xebia Cloud Financial Management cycle exists of five steps, grouped by Cost Analysis, Cost Optimization and Cost Management.



1. Gain visibility
Visibility of the generated costs by your cloud platform is a fundamental first step.

2. Understand the Costs
Allocate costs to services, applications, departments and more to help understand your costs.

3. Measure and Control
Measure your cloud costs and bring in the ability to take control.

4. Establish Benchmarks
Define realistic benchmarks that will help groups contribute (on every level) to specific and company-wide goals.

5. Drive Accountability
Make the application teams responsible and provide them the insights and tools to take ownership.

Analyze
Optimize
Manage

Introducing Cloud Financial Management

What makes cloud cost management so difficult?

The main reason for the lack of cost control is that cloud costs are complex. Cloud bills can easily grow up to thousands of cost line items. Service billing models are also diverse and complex. Each cloud service has specific characteristics and cost elements, sometimes up to tens of elements on an individual service.

Just parsing this huge amount of data is already a hard task. Thanks to the hard work of the billing teams within the cloud providers, we have that information parsed already, but the data is still not necessarily “consumable” (i.e., not everyone that has access to the data can make sense of it). Cloud Financial Management shows that with a disciplined and structured approach, you can become very successful at managing your cloud resources and controlling your expenses.

The continuous Cloud Financial Management framework.

At Xebia, we have introduced the Cloud Financial Management Cycle (left). This cycle is made with the purpose to implement cloud cost management within any organization. As with all cycles, we emphasize that after walking through it in its entirety, you end up at the start again. In this whitepaper, we will mainly focus on the middle: Measure & Control and Establish Benchmarks. This is where cost optimization really takes place and where we further explain our five perspectives on how to do it effectively.

Five Perspectives on Cost Optimization on the AWS Platform

In our cloud cost optimization initiatives, we analyze the AWS consumption from five perspectives which together give a holistic view on areas that are to be improved. In this whitepaper we detail out those perspectives in the context of the AWS platform. However, many of these perspectives and underlying concepts are universally applicable across other cloud platforms.

Perspective 1: Rightsizing & Elasticity

AWS resources such as EC2 Instances, EBS Volumes, Redshift and RDS Clusters are often overprovisioned and run constantly. Using AWS' inherent elastic features, we can match resources to demand to minimize our costs as well as environmental impact.

Perspective 2: Consumption Model Optimization

Up to 70% of cost savings are achievable by committing your capacity for 1 or 3 years on specific AWS services including EC2, RDS and Redshift. For spot instances, these savings can tally up to 90%. Consumption Model Optimization helps you choose the appropriate strategy.

Perspective 3: Service Characteristic Optimization

Each of AWS's 200+ different services has unique pricing characteristics. Analyzing your service-specific characteristics helps you identify misuse of a service from a cost efficiency perspective.

Perspective 4: Service Architecture Optimization

AWS has a wide assortment of native tools to answer your needs. Leveraging the right tools per specific use case in your environment helps you create an optimal architecture where technological, cost and usage advantages can be obtained.

Perspective 5: Service Pattern Analysis

Not all cost behaviors can be analyzed through standardized checks. Implementing a reliable Service Pattern Analysis methodology can help avoid unexpected deviations from your normal workload, as cost-patterns are easily identified and further analyzed.

Perspective 1

Rightsizing & Elasticity

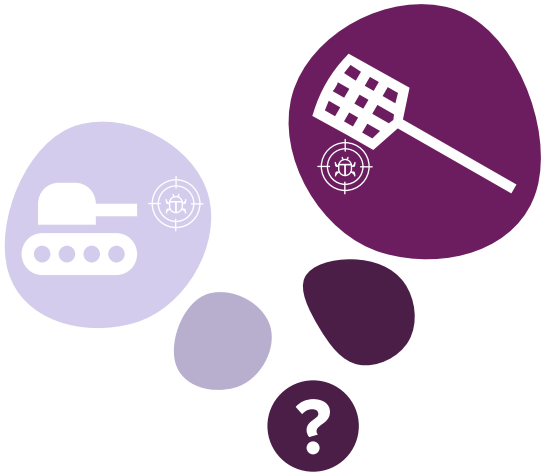
Before we start off going into the technical details of our five perspectives, let's first set the stage with a metaphor.

After their children have moved out of the house, parents often decide to move to a smaller house since they don't require the large space anymore. A main motivation is that this move significantly reduces their living costs. This financial decision is called rightsizing, and a similar strategy applies to the cloud too.

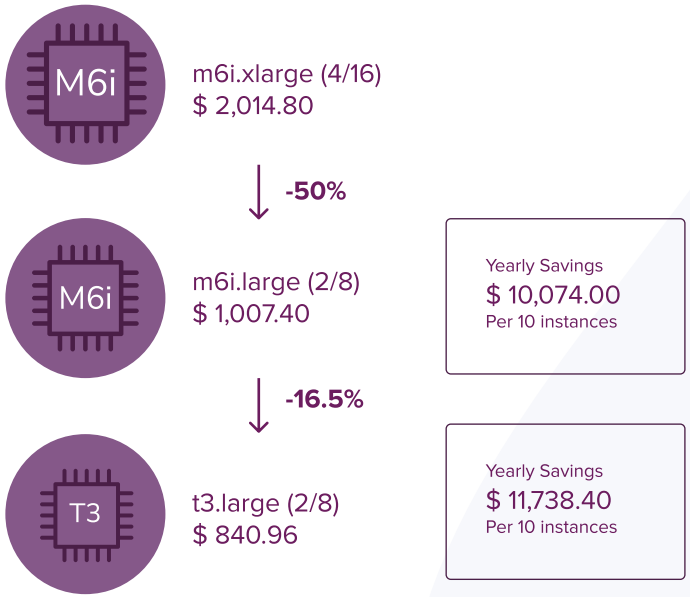
Rightsizing is all about reducing over-capacity of provisioned resources. We have a long history of leading datacenter-to-AWS migrations, and we learned that servers can often be provisioned up to 40% smaller than the current allocated capacity in the firm's datacenter. To put this statement into numbers we explain our actions based on actual AWS services and their prices. Imagine you're an AWS customer and you employ a m6i.xlarge instance (4 vCPU / 16 GB) for your on-demand workloads in Frankfurt. At current prices (2023), the yearly costs for this instance would be $2,014.80. These specifications would allow you to deploy your workloads comfortably and reliably, but is this luxury necessary?

This is where rightsizing starts. We analyze the workload that needs to be processed, and based on those results we could determine whether the m6i.large (2 vCPU / 8 GB) would be sufficient for the same workload. In this case, the cost drops by 50% to $1,007.40 on a yearly basis for this instance alone. And if it is not a production workload, we could even realize another 16,5% price drop by shifting to a burstable instance type. An example of a reliable and cost-efficient burstable instance would be a t3.large instance in Frankfurt, which costs $840.96 yearly.

In this example, we have rightsized a single instance. But let's say that you find ten instances within your organization where you can do this exercise: you would save more than $11K a year. And in our experience over the last few years, the majority of the organizations will have much more than ten comparable instances.



Service Rightsizing



Amazon Linux on demand EC2 instances in Frankfurt | April, 2023

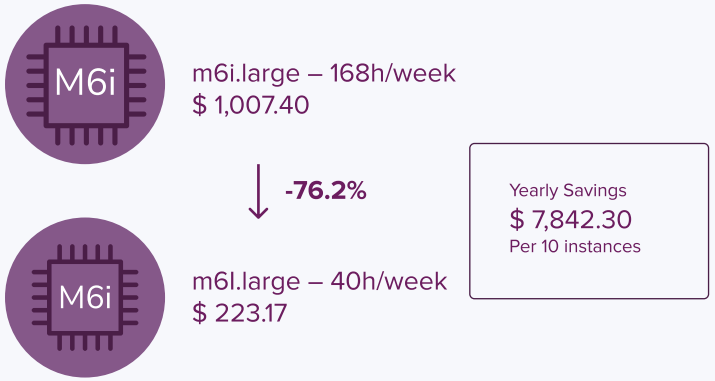
Zombie-assets

Another thing to pay attention to when talking about instance rightsizing is zombie-assets. It's common that organizations have resources running on AWS that they are not aware of. Usually because those resources don't have an owner anymore or are just not actively in use. But even if you don't know they are there, they will still generate costs. These resources do not only have to be compute instances but can also be storage related such as S3 buckets or objects, EBS volumes or DynamoDB tables. Make it a common practice in your organization to regularly review the need for certain resources and decommission them when possible and define pragmatic data life cycle management.

Apply elasticity to supply management.

Essentially, there is no need to provision for peak capacity all the time. For many applications, demand is not stable and may fluctuate hourly, daily, or seasonally. In this case, running all instances on peak capacity is wasteful from both a cost and a carbon emissions point of view. For example, some customer-facing applications may benefit from smaller instances in an autoscaling infrastructure to manage supply to the demand. For non-production environments, compute instances may be completely stopped outside of office hours. Running a compute instance for 40 hours a week instead of 24/7 would reduce the costs and carbon footprint by more than 75%.

Service Elasticity



Amazon Linux on demand EC2 instances in Frankfurt | April, 2023

Look at provisioned capacity regularly.

It's a good practice in supply management to re-evaluate provisioned capacity after a certain amount of time to make sure that whatever has been provisioned at a certain time is still accurate. For example, look at DynamoDB provisioned read/write capacity units.

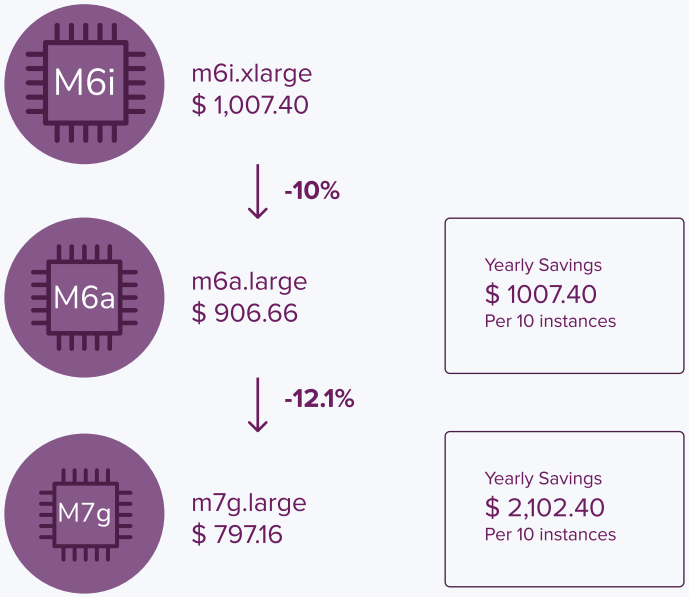
Optimize Computing Technology

When deciding on your computing technology, look at more efficient instance types besides the regular ones offered by AWS. Take for example the T instances. This line of instances is a great price-efficient alternative to the M instances, especially suitable for non-production workloads. Another consideration would be to look at instances that are non-intel based. AMD EPYC instances are incredibly well-optimized and could lead to another 10% cost savings compared to the Intel instance processors.

For further cost decrease on compute, you may want to consider whether your applications could run with ARM chips instead of x86 chips. AWS' ARM based Graviton processors run more efficiently, which could effectively reduce costs as well as carbon emissions. For example, a migration from m6i.large to m7g.large would save approximately 21% on costs. Another way to integrate Graviton is to select instance from the a1 family, which is a more compute-optimized type.

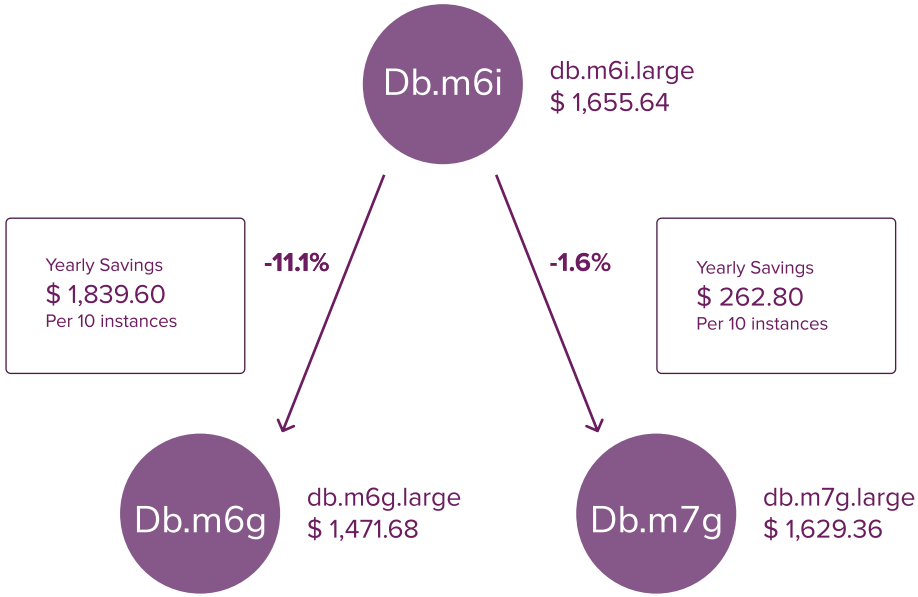
To go even further on the use cases for this type of optimization, we can consider the use of different processors on Relational Databases consumed via RDS, where the underlying infrastructure is an AWS responsibility. To exemplify, we can look at running MySQL db instances. Running a very common instance type, such as the db.m6i.large, that will deliver 2vCPUs, 8GiB of memory and up to 10 Gigabits of network performance has an on-demand yearly cost of $1,778.28. Switching it to a Graviton based db.m6g.large will bring exactly the same specifications, but at an yearly cost of $1,471.68, over 11% cheaper. And even the newer db.m7g.large, that will elevate the networking performance to up to 12.5 Gigabits will be cheaper than the m5, although just over 1.5%.

Processor Optimization - Compute Instances



Amazon Linux on demand EC2 instances in Frankfurt | April, 2023

Processor Optimization - Database Instances



MySQL on demand RDS DB instances in Ireland | April, 2023

Perspective 2

Purchase Model Optimization

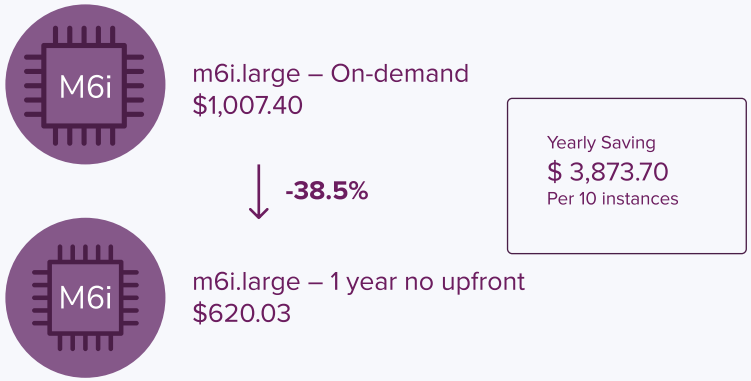
After determining your optimal instance size, the next step is to look at the second perspective: Purchase Model Optimization. To keep our theory close to the actual practice, we will once again dive into the AWS platform and its Reserved Instances.

Let's look again at the now familiar m6i.large instance type in the Frankfurt region. In the previous example, we used the costs associated with on-demand EC2 computing.

From a technical point of view, Reserved Instances offer the same compute performance with the same configurations. The biggest difference between traditional instances and reserved instances is that the period for which you want to use these instances is now fixed for a set amount of time, and optionally accompanied with upfront payment, partial or total. For example, if you're willing to commit for one year with all upfront payment, you can reduce the compute costs by over 38%.

And again, if you were to optimize ten instances of this size in the Frankfurt region in your workload, you would save more than $3,800 dollars a year. So even though AWS allows for a very flexible approach to selecting your instances, committing to, and reserving your cloud instances upfront can lead to a pretty serious saving.

Consumption Model Optimization



Amazon Linux EC2 instances in Frankfurt | April, 2023

Things to consider

- Reserved Instances (Up to 75%)
- Spot Instances (Variable, up to 90%)
- Savings Plan (25% - 57%)
- Workspaces subscription (~85% monthly vs hourly)
- Enterprise Discount Programs (7% - 15%)
- Reseller (value added) propositions.

Savings Plans.

While we are on the topic of Purchase Model Optimization, we propose some other areas to look at in Perspective 2. For instance, saving plans. We touched on reservations, but AWS also has another model available as of November 2019: the Savings Plan. If you are a frequent user of on-demand workloads, look at the available committed savings plans and continue to reduce your instance costs. The savings plan, in its compute modality, also apply to services such as AWS Fargate and Lambda.

Spot Instances.

As described by AWS, Spot Instances let you take advantage of unused EC2 capacity in the AWS cloud. Spot Instances are available at up to a 90% discount compared to on-demand prices. This type of instance is very useful for test workloads or workloads that can deal with a sudden termination, in that case also in production environments. When application architecture lends itself towards it, for fault tolerant and stateless applications, spot instances can decrease running costs rather dramatically. Use cases include API endpoints, analytics applications, batch processing tasks, and high-performance computing.

Purchase Subscription Type.

For some services, like Amazon Workspaces for example, it's wise to look at the monthly model versus the hourly model, as it is often the case that the monthly model is a much more affordable option.

Bring-Your-Own-Licensing models

If you spend more than millions a year on the AWS platform, consider going for an enterprise discount program agreement. Especially if you intend to sign up for other licenses from software vendors via the AWS Marketplace.

Value Added Resellers

Based on years of experience in reducing and optimizing cloud spend in various industries, we offer Cost Management services as part of our added value resell services in AWS. More information on our managed savings and cost scan services can be found here.

AWS Service	Reservations	Saving Plan	Spot
EC2	🟢	🟢	🟢
Fargate	🔴	🟢	🟢
Lambda	🔴	🟢	🔴
RDS	🟢	🔴	🔴
Redshift	🟢	🔴	🔴
Opensearch	🟢	🔴	🔴
ElastiCache	🟢	🔴	🔴

* Purchase type options for services mentioned before.

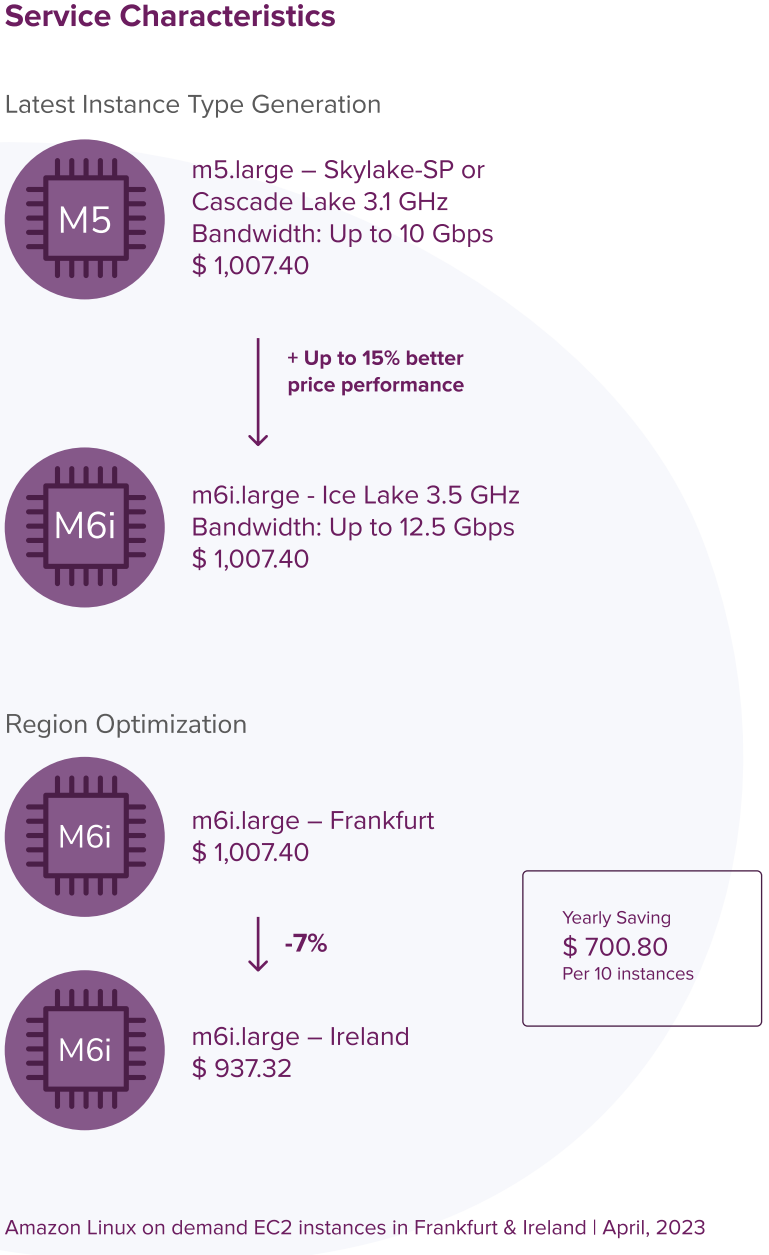
Perspective 3

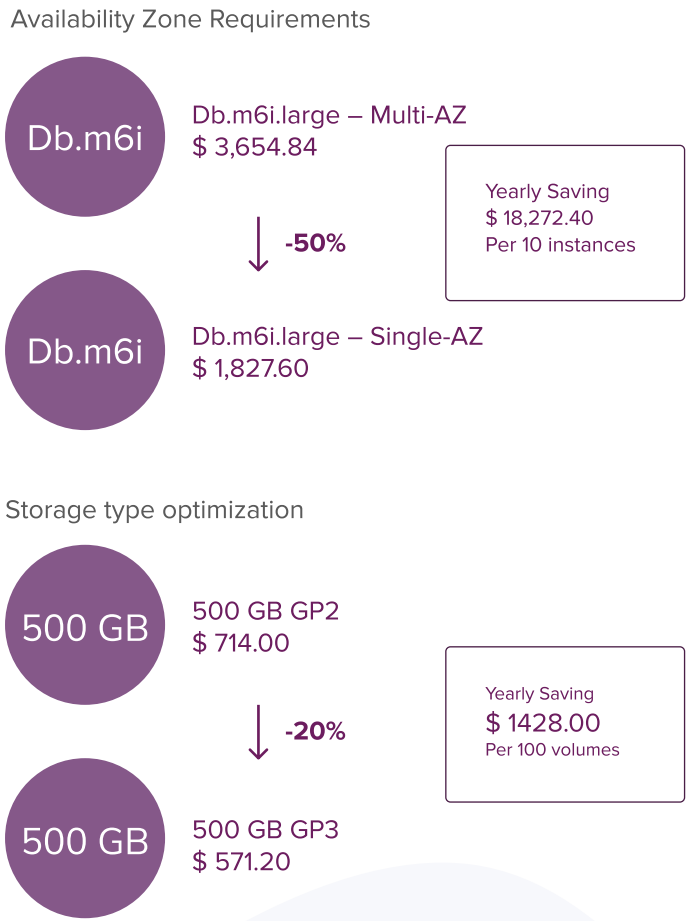
Service Characteristic Optimization

Let's move on to Perspective 3: Service Characteristic Optimization. Are you using the AWS service according to the latest cost-effective best practices? AWS releases new generations of instances every year, each more price efficient than the last. Therefore, we advise you to regularly review whether you would benefit from upgrading to these new instance generations when they become available.

A migration from a M5 series instance (older gen) to a M6i series (next gen) would lead to better CPU performance and a larger bandwidth for the same price, thus lowering performance per dollar for these instances. For instances running with an AMD processor, this migration would yield similar benefits whilst also reducing the per hour.

Another way to increase instance efficiency would be through region optimization. The region that you choose to run your cloud workload can also influence your cloud bill. When comparing the same provisioned workload in Frankfurt to Ireland as an alternative, you would also approximately 7% on the resource cost for the same performance. This can be explained by the difference of local costs in different regions (labor, electricity, housing). We advise you to extensively research which region is best suited for your operations and optimize accordingly.





Amazon on demand RDS MySQL instances & EBS volumes in Frankfurt | April, 2023`

Availability Zone Requirements

If you use AWS RDS databases, you can deploy them as “highly available” in a multiple availability zone setup. This feature creates a backup database and thus directly doubles the costs. For critical production workloads this is a common practice. However, for Dev/Test/Acceptance workloads and even some other production workloads this may not be a strict requirement. You can reduce a serious amount of cost by provisioning those workloads in a single availability zone if this suits your organizational requirements.

Storage Type Optimization

Another important consideration to make is whether your provisioned storage type matches your storage needs properly in Amazon EBS. For high performance block storage requirements, provisioned IOPS (IO2) is a great choice, with a maximum IOPS of 256000 and a maximum throughput of 4000 MB/s. Most applications, however, do not require this and can function with the more competitively priced general-purpose SSD (GP3). This switch can realize cost savings of up to 70%.

Another way to drive down EBS costs would be to switch to the newest generation, from IO1 to IO2 or from GP2 to GP3. Take the general-purpose SSD types for example, the switch from GP2 to GP3 yields a higher base performance and reduces costs by 20%.

More to consider

These examples point out how we look from this perspective but there are many more examples. If you’re running massively in AWS at least dive into the following to consider as well:

- Latest instance type family
- Storage types / storage tier
- Multi AZ deployment Evaluation
- RDS Storage autoscaling
- Region optimization
- DynamoDBIndex optimization
- Scalability
- TTL and Cache optimization

Perspective 4

Service Architecture Optimization

Service Architecture Optimization – are you using the most appropriate and most efficient service for your use case?

Let's consider a containerized web application on a Kubernetes cluster that receives moderate variable traffic, with a peak utilization of 2 vCPUs and 8 GB over the year and an average utilization of 1 vCPU and 4 GB. The company has various options to create an architecture for this application.

The first option is running Amazon Kubernetes on an m6i.large EC2 instances, which would cost $1,906.20.

If you were to migrate this container workload to an alternative container service model, in for example Fargate, it significantly decreases the cost level by more than 400 dollars, due to the fact that this service is better equipped to handle the variability in traffic. Continuing, if you don't specifically require Kubernetes, you can look at the native container service of AWS (Amazon ECS on Fargate). It decreases the costs again with almost an additional 900 dollars. Notably, this scenario demonstrates that a managed service, which in any case decreases the operational overhead, may also be beneficial to the bottom line of the AWS bill.

A similar example of service architecture optimization can be found in the OS levels that are available to AWS customers. If you are working with RedHat, transitioning to Amazon Linux would save almost 35% of your EC2 spend for the same instance performance. This is effectively the license cost of RedHat, which is not applicable in Amazon Linux. This makes paying attention to you OS of choice an important area to look at.

Service Architecture Optimization

Container Workload Optimization

EC2

Amazon EKS on EC2
m6i.large + 20 GB GP3
$ 1,906.20

vCPU

1 vCPU + 4 GB
$ 1,462.92

vCPU

Amazon ECS on Fargate
1 vCPU + 4 GB
$ 586.92

OS Optimization

M6i

m6i.large – Redhat Linux
$ 1,533.00

↓ **-34.3%**

M6i

m6i.large – Linux
$ 1,007.40

Yearly Saving
$ 5,256.00
Per 10 instances

Amazon Linux on 86x in Frankfurt | April, 2023

A few other quick tips to explore within Service Architecture Optimization:

- To reduce your networking cost, look at VPC endpoints to make more efficient use of your traffic.
- Consider Amazon Glacier for archive storage instead of S3.
- Look at optimizing your network routing.
- Invest in Code/Query optimization: this covers all kinds of areas that are affecting your application architecture but that are also open to significant savings.

Things to consider

- VPC endpoints
- Apply caching
- Build-in elasticity
- Offloading logging data
- Fargate for containers
- Open Source database engine
- Amazon Linux
- Glacier vs. S3 for archives
- Routing change
- Code / Query optimization
- FSx vs. ec2 hosted fileserver

Perspective 5

Service Pattern Analysis (Deviating patterns)

In this last part we will give you our best practices for the fifth cost optimization perspective: Service Pattern Analysis. This concerns conducting thorough analysis on your cloud spend as well as actively slicing and dicing.

From this perspective, it is valuable to look at abnormal behavior, such as unexpected peaks or deviations from the expected trend line. This allows you to identify costs and behaviors that you would not find in checklists and frameworks.

Service Pattern Analysis – Tips

1. Measure baseline performances, tag resources and collect metrics.
2. Set bandwidth for 'normal behavior', automate breaches.
3. Involve the right people, resources should have owners.
4. Prioritize investigating priorities, have a clear escalation path.
5. Remediate and prevent, perform a root cause analysis

Take two practical examples:

1. Financial Anomaly

A customer employs an application that is using an S3 bucket with a monthly spend of $300. So far nothing is out of the ordinary, right? However: that same bucket had generated a cost level of 3.5k dollars a month for API calls alone.

This is financial anomaly, especially compared to the size of the storage. When we started to investigate it together with the customer's application team, we found that the code was not cost-efficient. More specifically, the code was pulling the bucket every minute, which was not a customer requirement. Since the number of requests is a cost driver in S3, we prevented further overspending by optimizing the application code and changing the pulling method.

2. Hacked Behavior

Another example is an organization that had a website with a stable and predictable user base. However, the autoscaling group was blowing up at the backend. When we were asked to investigate this, we found that, through an analysis of the sources of the costs, that the website had actually been hacked.

The Amazon Machine Images (AMIs) used by the autoscaling group contained crypto miners. We were able to figure this out by receiving the warning at the cost level. Thanks to this early discovery through actively monitoring costs and responsible channels, a lot of damaged was prevented. But of course: don't rely on financial management tools alone for your security.

Conclusions & Takeaways

- **Start with cost insights and awareness**

Cost optimization starts with visibility. Drive accountability by providing insights and make sure all AWS stakeholders in your organization are aware and have the overview of the costs that they are generating on the AWS platform.

- **Focus on the quick wins (rightsizing, zombie asset reduction, reservations)**

When you start with optimizing, focus on the big wins. Follow the order elasticity, rightsizing, and then commitments. These are all easy adjustments to immediately make a difference on your cloud bill.

- **Approach Cloud Cost management from TCO perspective**

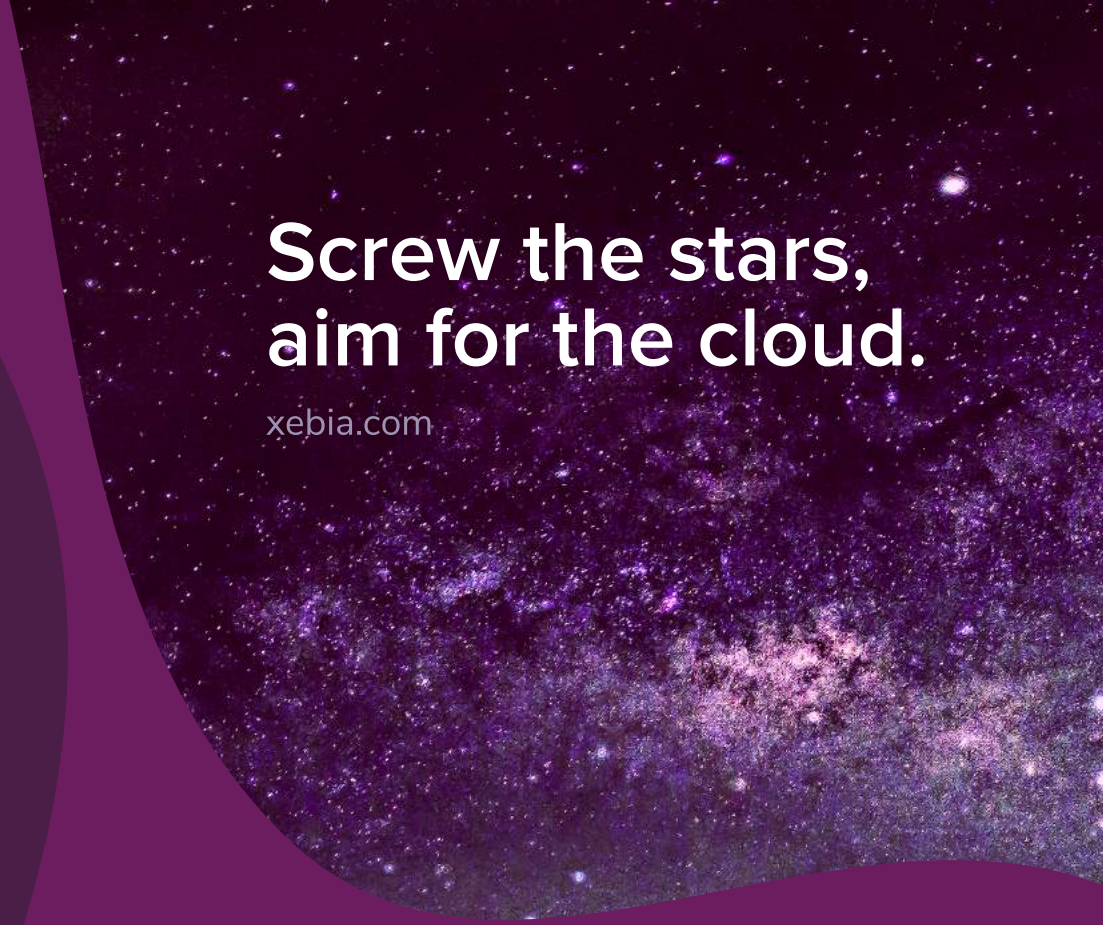
Approach cloud cost management from a TCO perspective and don't base it solely on reducing the AWS bill directly.

- **Drive the full cloud cost management cycle**

Make sure that you continuously drive the circle and maintain your improvements. It is a learning process for you and your team members. But, by making it a standard practice, it does become easier.

- **Request help, there is always a business case for cost optimization**

Last but not least: request help if you are struggling with these cost management applications. As you have read in the paper, there is always a business case to be made regarding cost optimization. For every AWS service, there is a best practice and opportunities to prevent unnecessary cloud spending.



Xebia Identity

Xebia explores and creates new frontiers. Always one step ahead of what businesses need, we turn the latest technology trends into advantages for our customers. As a mainstream frontrunner, we create new solutions and build the future with our clients.

An international network of passionate technologists and pioneering craftsmen, Xebia provides the cutting-edge tools, training and consulting services that make businesses work better, smarter, and faster.

Xebia