The Serverless Pricing Model

In this lesson, you will learn about the pricing model of AWS serverless architecture.

WE'LL COVER THE FOLLOWING

Lambda pricing

Technically, AWS Lambda and similar systems are supercharged container management services. They provide standardized execution environments to activate applications very quickly, and algorithms to automatically scale containers according to the workload. Although running those services is technically challenging, this is just an incremental improvement on a decade-long journey towards application virtualization. That's why technical architects (especially those who built and operated their own container clusters in large companies) sometimes complain about how serverless is just a marketing fad. The revolutionary part of Lambda is something that other cloud providers are quickly copying: the financial side of the story. The serverless pricing model is a lot more important than the technology for application developers.

Reserved capacity

When using AWS Lambda to run code, you pay for *actual usage*, not for *reserved capacity*. If the application isn't doing much, you don't pay for anything. If millions of users

suddenly appear, Lambda will spin up containers to handle those requests, charge you for doing so, and then remove them as soon as they are no longer necessary. You never pay for idle infrastructure or when tasks are waiting on user requests.

> Reserving minimum capacity
>
> In December 2019, AWS enabled users to reserve minimum capacity for Lambda functions, ensuring that there is always a certain number of processes waiting on user requests. In AWS jargon, this is called *Provisioned Concurrency*. With provisioned concurrency, you will also pay a fixed price for reserved instances, regardless of whether they are used or not. However, most applications won't need to use this feature, if you design them well.

Lambda pricing depends on two factors:

1. The maximum memory allowed for a task.
2. The time it spent executing for a task.

As an illustration, assuming a configuration with 512 MB allowed memory, AWS charged the following fees for Lambda in the USA:

- $0.0000002 per request.
- $0.000000833 for running 100ms with 512 MB of working memory.

Comparing reserved virtual machine pricing while paying for utilization isn't straightforward because they depend on different factors. Reserved capacity pricing usually depends on the expected load, and utilization pricing depends on the nature of tasks being executed. Here are two examples of various extremes.

Comparison with AWS Elastic Compute Cloud (EC2)

For an infrequent task, say something that needs to run every five minutes for about 100 ms and needs 512 MB of memory, Lambda pricing would roughly work out to slightly less than 1 US cent a month. Renting a similar hardware configuration from AWS Elastic Compute Cloud (EC2), assuming there is a primary and a fail-over virtual machine in case of problems, would cost roughly 9 USD; three orders of magnitude more.

> Note that for Lambda, it's not necessary to reserve a backup system in case the primary one fails; this is already provided by the platform and is included in the price.

At the other extreme, a single Lambda function continuously receiving requests and never stopping, over a month, would cost roughly 27 USD. Just looking at basic hosting costs, reserving virtual machines seems cheaper. But this is only if you ignore all the operational services that are included with AWS Lambda and assume that a single virtual machine is enough to handle this sustained load. For something getting continuously hit over a long time, it's much more likely that it would run on a whole cluster of load-balanced machines with several backup systems waiting in reserve. Working with virtual machines requires operations experts to plan capacity carefully, predict load, and automate container cluster scaling to match expected demand. With Lambda, all that is included in the price, as well as recovery from errors, logging, monitoring, and versioning.

The difference between billing for reserved capacity and actual usage is also important for testing and staging environments. When companies pay for reserved capacity, copies of the production environment usually multiply the operational costs, even though they are idle most of the time. That is why staging and acceptance testing environments usually end up being slimmed-down versions of the real thing. With billing based on actual usage,

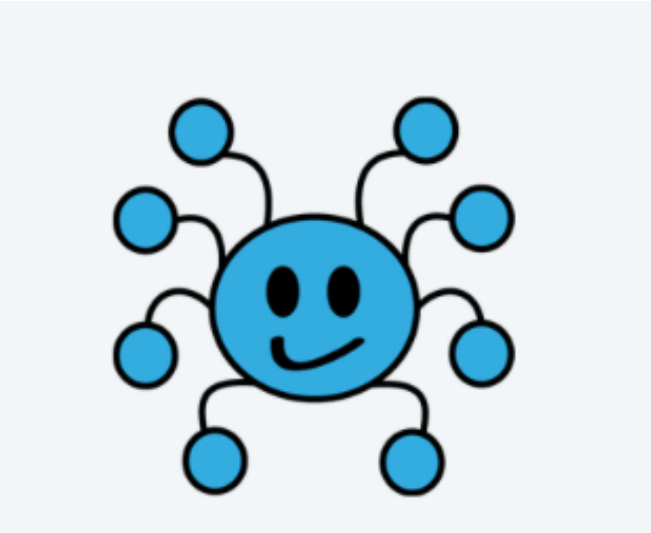
environments don't cost anything if they are idle, so the economic benefits of

maintaining separate slimmed versions disappear. For most organizations, testing environments with serverless architectures are effectively free.

MindMup

For serverless applications, the provider controls the infrastructure, not the application developers. This means that developers can focus on things that make their application unique (the core business logic) and not waste time on operational or infrastructural tasks.

* When moving MindMup from a hosted environment based on virtual machines to AWS Lambda, we realised that we could drop a lot of source code that performed infrastructural tasks, and since late 2016 have not needed to write any serious infrastructural code at all. We didn't have to spend time building and integrating monitoring and scaling systems or worry about most operational issues. Lambda helped us go from a conceptual idea to getting our working software in front of users significantly faster.



In the next lesson, you will learn about the effects of request pricing on deployment architecture, security, and product decisions.