Compute: Lambda

Lambda service as a code runner in the cloud or a plugin service for other AWS services is discussed in this lesson together with our rule of thumb in using Lambda service.

We'll cover the following

Code runner in the cloud

If EC2 is a complete computer in the cloud, Lambda is a code runner in the cloud. With EC2 you get an operating system, a file system, access to the server's hardware, etc. But with Lambda, you just upload some code and Amazon runs it for you. The beauty of Lambda is:

- It's the simplest way to run code in the cloud.
- It abstracts away everything except for a function interface, which you get to fill in with the code you

want to run.

Misuse of Lambda

We think Lambda is great—definitely one of the good parts of AWS—as long as you treat it as the simple code runner that it is. A problem we often see is that people sometimes mistake Lambda for a general-purpose application host. Unlike EC2, it is very hard to run a sophisticated piece of software on Lambda without making some very drastic changes to your application and accepting some significant new limitations from the platform.

Lambda as a plugin system for other AWS services

Lambda is most suitable for small snippets of code that rarely change. We like to think of Lambda functions as part of the infrastructure rather than part of the application. In fact, one of our favorite uses for Lambda is to treat it as a plugin system for other AWS services. Let’s look at a few examples of it.

S3

- S3 doesn’t come with an API to resize an image after uploading it to a bucket, but with Lambda, you can add that capability to S3.

Application Load Balancer (ALB)

- Application load balancers come with an API to respond with a fixed response for a given route, but they can’t respond with an image. Lambda lets you make your load balancer do that.

CloudFront

- CloudFront can’t rewrite a request URL based on request cookies (which is useful for A/B testing), but with Lambda, you can make CloudFront do that with just a little bit of code.

CloudWatch

- CloudWatch doesn’t support regex-based alerting on application logs, but you can add that feature with a few lines of Lambda code.

Kinesis

- Kinesis doesn’t come with an API to filter records and write them to

DynamoDB, but this is very easy to do with Lambda.

CloudFormation

- CloudFormation's native modeling language has many limitations and, for example, it can't create and validate a new TLS certificate from the AWS Certificate Manager. Using Lambda, you can extend the CloudFormation language to add (almost) any capability you want.

And so on—you get the idea. Lambda is a great way to extend existing AWS features.

Limitations

Treating Lambda as a general-purpose host for your applications is risky. It might look compelling at first—no servers to manage, no operating system to worry about, and no costs when unused—but Lambda's limitations are insidious hidden risks that typically reveal themselves once your application evolves into something bigger.

Then, things start coming at you one after the other, and before you know it, you are spending most of your time trying to figure out convoluted solutions to Lambda's numerous limitations.

Q

Unlike S3, it is very hard to run a sophisticated piece of software on Lambda without making some very drastic changes to your application.

Retake Quiz

Short term limitations

We hope that some limitations will likely improve or go away over time. Let’s look at a few examples.

Cold start

A very annoying issue is the cold start when a function is invoked after a period of inactivity or when Lambda decides to start running your function on new backend workers.

Limit on code bundle

Another problem is the limit of 250 MB for your code bundle, including all your dependencies. Depending on the programming language you’re using, you can find yourself quickly exhausting this limit. And the network bandwidth from Lambda functions seems to be very limited and unpredictable. These can all be problematic, depending on your use case, but we’re quite confident that these issues will improve over time.

Inherent Limitations

As opposed to short term limitations that we expect to go away, there are other limitations that are inherent to the way Lambda works and which are less likely to go away.

Stateless

- For example, you have to assume that every Lambda invocation is stateless. If you need to access some state, you have to use something like S3 or

DynamoDB.

While this works fine for a demo, it can quickly become prohibitively expensive in the real world.

- For example, handling a WebSocket connection on Lambda will likely require a read and write to DynamoDB for every exchanged packet, which can quickly result in a spectacularly large DynamoDB bill, even with modest activity.

↓ RULE OF THUMB ↓

If you have a small piece of code that will rarely need to be changed and that needs to run in response to something that happens in your AWS account, then Lambda is a very good default choice. You can even define the code in your CloudFormation template so that this piece of custom functionality truly becomes part of your infrastructure. For everything else, we advise a lot of caution.

> That said, we are very optimistic about the future of serverless computing. The idea of abstracting away everything in the stack beneath your code is a phenomenal advance in software development. However, when building software in the present, we have to assess the options available to us today, and while Lambda has its place, it is certainly not a substitute for EC2.

In the next lesson, we will take a look at ELB and it’s different attributes.