# What is Lambda?

AWS Lambda lets you run code without provisioning or managing servers. A Lambda code executable, called a function, can be set up to automatically trigger from other AWS services or call it be called directly from any web or mobile app.

Lambda is one component of Amazon’s “serverless” technology\* ecosystem. Other components may include API Gateway, SQS, Dynamo DB (BaaS), etc. Lambda may also be referred to as function-as-a-service (FaaS) or nanoservices architecture.

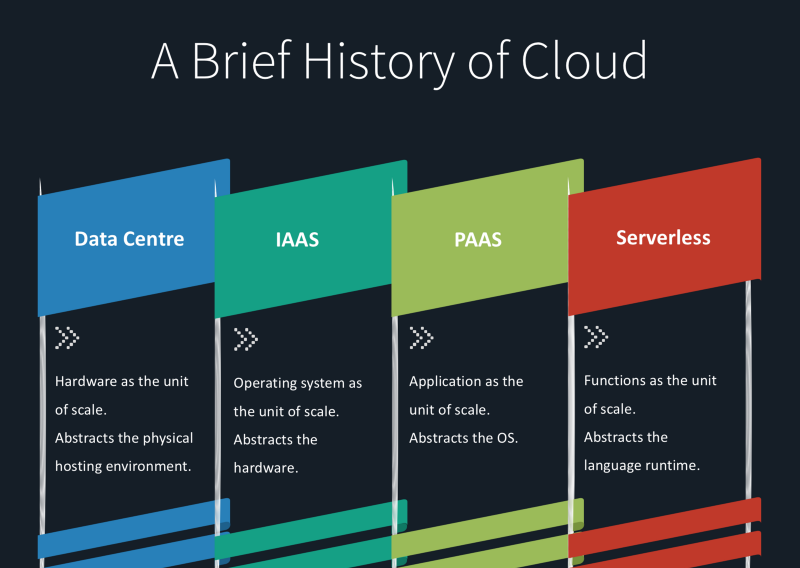
<https://aws.amazon.com/lambda/>

\* Lambda is often used as a synonym for serverless architecture/technology.

## 

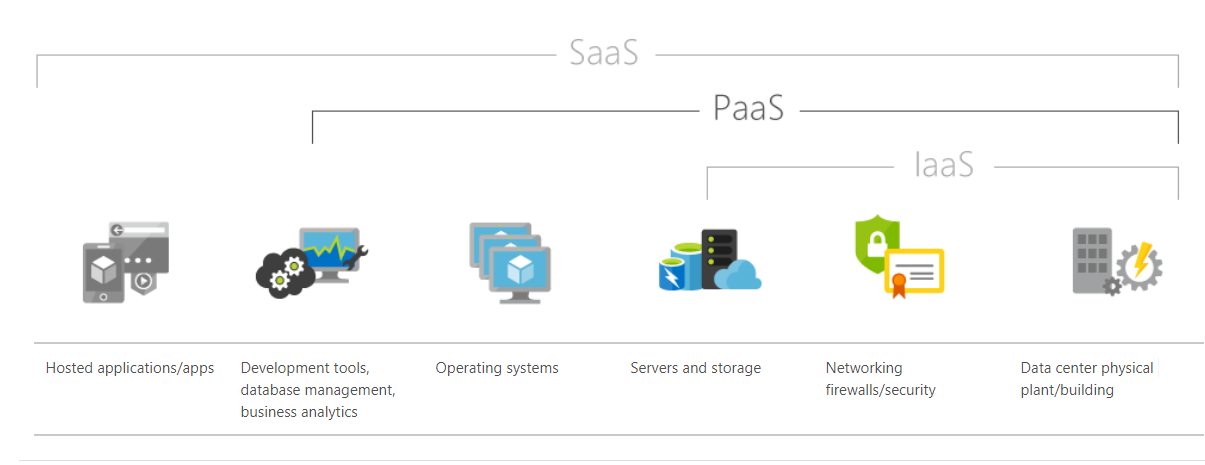
<https://www.slideshare.net/mitocgroup/serverless-microservices-real-life-story-of-a-web-app-that-uses-aws-lambda/18>

## Serverless vs. Platform-as-a-Service



<http://www.aurosyssolutions.com/cloud-computing/what-is-serverless-computing/>

A PaaS provider builds and supplies a resilient and optimized environment on which users can install *applications* and data sets. In contrast, the unit of abstraction is the *function* for serverless architectures instead of an “always-on” application.



<https://azure.microsoft.com/en-us/overview/what-is-paas/>

# Why Lambda?

AWS Lambda is a “serverless” technology\* which allows the developer to focus on building and executing code.

* “A decade ago, cloud servers abstracted away physical servers. And now, ‘Serverless’ is abstracting away cloud servers.” – [Adam Watt](https://medium.freecodecamp.org/going-serverless-how-to-run-your-first-aws-lambda-function-in-the-cloud-d866a9b51536)
* “You stop [shaving yaks](http://www.hanselman.com/blog/YakShavingDefinedIllGetThatDoneAsSoonAsIShaveThisYak.aspx), and start buying hairless yaks.” – [Ben Kehoe](https://serverless.zone/serverless-vs-paas-and-docker-three-legged-hairless-yaks-f2c4191b1ed6)

<https://aws.amazon.com/serverless/>

You pay by the execution count as well as the duration of execution (based on the memory allocation). You will also be charged for transferring date, e.g. reading from/writing to S3.

Unless you know your workload will definitively result in a lower monthly cost with static instances, using serverless technologies is a good first step when creating applications.

<https://aws.amazon.com/lambda/pricing/>

Since serverless technology lowers the administrative concerns with infrastructure, using Lambda can result in operational cost savings. Lambda does not mean NoOps, but perhaps LowOps.

# Why not Lambda?

Some say a serverless system is less reliable. Usually what this is referring to is that there are more points of failure.

There may be a perception that a serverless system isn’t as full-featured, usually this is due to the narrower focus of the functionality.

* “That hairless yak? It may only have three legs.” – [Ben Kehoe](https://serverless.zone/serverless-vs-paas-and-docker-three-legged-hairless-yaks-f2c4191b1ed6)

Higher per-compute-cycle cost of FaaS is going to make traditional architecture more attractive for predictable loads.

Systems can be difficult to architect, especially with limited experience on what works and what may not be well-suited to Lambda. Growing interdependencies need to be understood.

Converting an existing system to Lambda typically requires a serious rearchitecting effort.

# Running Lambda

“Lambda is the glue that binds it all together.” – [Jeremy Edberg](https://www.youtube.com/watch?v=OI_V6OZZkZM) in reference to the AWS serverless ecosystem which includes database, SQS, etc.

Getting Started - <https://www.youtube.com/watch?v=OI_V6OZZkZM>

<https://cloudncode.blog/2017/03/02/best-practices-aws-lambda-function/>

In the case of AWS Lambda Functions, something that invokes execution is called a **trigger**. A trigger may be an HTTP request, a new document upload to S3, a scheduled event, a notification from AWS Simple Notification Service (SNS), etc.

Lambda can be built using Node.js, Java, Python or C#.

<http://docs.aws.amazon.com/lambda/latest/dg/programming-model-v2.html>

For Node.js, a callback is used to return a result. The result provided must be JSON.stringify() compatible. Other languages are similar. Results must boil down to a JSON string. E.g. In Java, the return object will be serialized to JSON.

<http://docs.aws.amazon.com/lambda/latest/dg/nodejs-prog-model-handler.html>

# Demo

<http://docs.aws.amazon.com/lambda/latest/dg/get-started-invoke-manually.html>

<https://medium.freecodecamp.org/going-serverless-how-to-run-your-first-aws-lambda-function-in-the-cloud-d866a9b51536>

AWS Chalice - <https://www.youtube.com/watch?v=gZK3u2q5K5E>

# Lambda How-To

## Tips & Tricks

* Use immutable data when possible.
* Moving data around is the most expensive piece of a distributed system.
* Use queues. Not only does this create some resiliency, under the hood, AWS allows higher throughput from SNS or SQS.
* Limit your function size. Startup or initial load time is important (using containers).
* Functions are run in parallel (except step functions).
* The container may stay around for a bit; use this to your advantage.
* Setup alarms right away on CloudWatch.
* Keep an eye out for infinite loops. Check the callstack.
* Let Amazon know if you are going to do some massive scale up.

## Additional Resources

<https://d0.awsstatic.com/whitepapers/optimizing-enterprise-economics-serverless-architectures.pdf>

<https://aws.amazon.com/solutions/case-studies/thomson-reuters/>

<https://www.youtube.com/watch?v=AcGv3qUrRC4>

<http://docs.aws.amazon.com/lambda/latest/dg/monitoring-functions-logs.html>