FAAS VS SERVERLESS: WHAT'S THE DIFFERENCE?

Serverless and FaaS (function-as-a-service) are two new categories in the vast, ever-changing world of cloud computing services. Both may be useful in saving money, refocusing developers' time, relegating infrastructure management, and harnessing cloud technology.

Though the terms of FaaS and serverless are sometimes used interchangeably, they are not the same thing. This article breaks down the similarities and difference between FaaS and serverless, so you can determine if one, or both, is right for your company.

Serverless: why the confusion?

First up, let's clear the confusion around these terms, at least as much as possible. Serverless can be a tricky word to define, as with any new tech term, but also because "serverless" stands for lots of things.

"Serverless" may invoke an image of a completely server-free setup, which sounds dreamy but just isn't possible. Instead, serverless likely means using off-site servers to handle some of your computing burden, like cloud services that are housed in giant data warehouses around the world. As such, serverless can be best understood as "out of sight, out of mind".

Still, serverless can encompass more than one technology. Depending on the context, "serverless" may refer to serverless computing, serverless architecture, or even serverless coding. For the purpose of this article, we are primarily talking about serverless computing as a system approach.

What is serverless?

In a traditional computing environment – aka not serverless – the central server application manages all flows, controls, and security.

In this new serverless version, however, no central authority balances these branches. Instead of orchestration, serverless environments emphasize choreography. Individual components have a more architecturally-aware role, not unlike that in microservices, which can make the application more flexible and amenable to change, a key benefit for many of today's businesses.

Generally, serverless computing is separated into two types, though there is some overlap. The first type comprises apps that use third-party, cloud-hosted services to manage server-side logic and state. These apps typically use a large ecosystem of databases and authentication services that rely on the cloud. These are generally known as backend-as-a-service, or BaaS.

What is FaaS?

The second and newer type of serverless computing is function-as-a-service. FaaS is a category of cloud computing services that is disrupting the way applications and systems have been built for decades, which is why it's driving the buzz around serverless now in a way that BaaS hadn't quite achieved.

In FaaS, the server-side logic remains in the app developer's responsibility, as in traditional architectures, but the server-side logic actually runs in stateless compute containers. These containers are triggered by an event, often ephemeral (lasting for only one innovation), and managed fully by the third-party vendor. Widely used FaaS options include AWS Lambda, Google Cloud Functions, Microsoft Azure Functions and open-source options from Oracle and IBM.

Like other as-a-service offerings, FaaS is a third-party platform that you only pay for when you use it. This platform can allow developers to focus on the design, running, and management of application functionalities without having to focus on building and maintaining backend infrastructure. It is precisely this reason that many developers embrace FaaS: to focus more on the end product and less on the setup that's inherent in traditional application design.

Plus, FaaS doesn't require devs to code to a specific framework or library, as the third-party service knows how to handle that. Deployment in FaaS is different from traditional architecture too – instead of running server apps, frontend code is uploaded to the service and the provides takes care of the backend nitty-gritty, like provisioning resources, instantiating virtual machines, and managing processes.

Choosing FaaS and serverless

You don't have to make a choice between one or the other, but beware that the decisions aren't apples-to-apples. An important distinction is that your IT team may employ FaaS without using a serverless architecture. In fact, a survey from August 2018 indicated that more than half of companies using FaaS options are not incorporating serverless computing.

FaaS can offer more narrow solutions, so that you can replace part or all of an app with an FaaS functionality that runs within an event-driven or HTTP context. This ease of deployment and agility means you can opt for the benefits of FaaS, particularly in specific use cases or test environments,

without the entire overhaul that may accompany a truly serverless approach. On the flip side, FaaS can increase computing costs when you scale up for larger tasks like production.

Serverless computing offers far greater functionality than smaller, pay-per-use FaaS options, particularly because it can encompass a variety of technologies. Cost-wise, serverless options may offer some savings, though it's not guaranteed.

Flexibility is a tradeoff for more complicated moving parts. Serverless tends to be more complicated than FaaS, as you're likely running more components, perhaps at a larger scale. While giving up some of this control can be freeing, you will still have to monitor that the dozens of moving pieces are aligning, a problem you likely didn't have in your traditional monolith system.

The flexibility you'll get and the cost you may save (if you're careful) might be worth the additional complexities – but understanding that FaaS and serverless aren't the same thing is the vital first step towards choosing the right solution.