SERVERLESS

Serverless Architecture Overview

Discover how adopting a serverless architecture can help you scale your applications in a cost-efficient way.

ON THIS PAGE

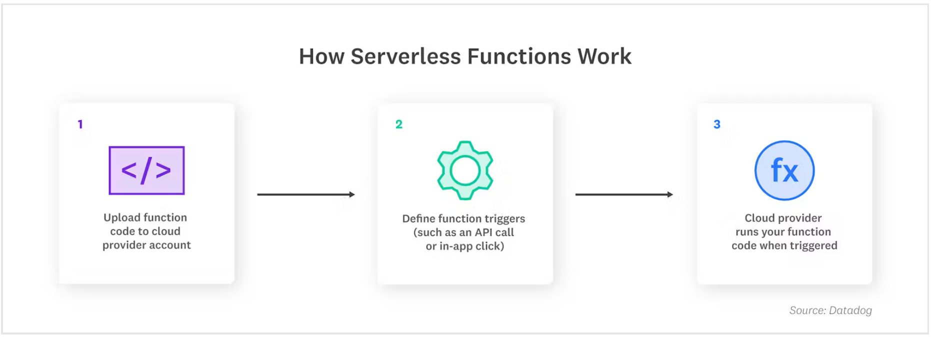
What is Serverless Architecture?

Serverless architecture is an approach to software design that allows developers to build and run services without having to manage the underlying infrastructure. Developers can write and deploy code, while a cloud provider provisions servers to run their applications, databases, and storage systems at any scale. In this article, we'll cover how serverless architecture works, the benefits and drawbacks of using it, and some tools that can help you go serverless.

How Serverless Architecture Works

Servers allow users to communicate with an application and access its business logic, but managing servers takes considerable time and resources. Teams have to maintain the server hardware, take care of software and security updates, and create backups in case of failure. By adopting serverless architecture, developers can offload these responsibilities to a third-party provider, enabling them to focus on writing application code.

One of the most popular serverless architectures is Function as a Service (FaaS), where developers write their application code as a set of discrete functions. Each function will perform a specific task when triggered by an event, such as an incoming email or an HTTP request. After the customary stages of testing, developers then deploy their functions, along with their triggers, to a cloud provider account. When a function is invoked, the cloud provider either executes the function on a running server, or, if there is no server currently running, it spins up a new server to execute the function. This execution process is abstracted away from the view of developers, who focus on writing and deploying the application code.


How Serverless Functions Work
1
Upload function code to cloud provider account
2
Define function triggers (such as an API call or in-app click)
3
Cloud provider runs your function code when triggered
Source: Datadog

Function as a Service (FaaS), a popular type of serverless architecture, allows developers to focus on writing application code.

While serverless architecture has been around for more than a decade, Amazon introduced the first mainstream FaaS platform, AWS Lambda, in 2014. Currently, a majority of developers still use AWS Lambda to build serverless applications, but Google and Microsoft have their own FaaS offerings as well, called Google Cloud Functions (GCF) and Azure Functions respectively.

Fundamental Concepts in Serverless Architecture

Although serverless architecture eliminates the need for server management, there's still a steep learning curve, especially if you're chaining multiple functions together to create complex workflows in an application. It can therefore be helpful to familiarize yourself with these fundamental serverless terms:

1. **Invocation**

A single function execution.

2. **Duration**

The time it takes for a serverless function to execute.

3. **Cold Start**

The latency that occurs when a function is triggered for the first time or after a period of inactivity.

4. **Concurrency Limit**

The number of function instances that can run simultaneously in one region, as determined by the cloud provider. A function will be throttled if it exceeds this limit.

5. **Timeout**

The amount of time that a cloud provider allows a function to run before terminating it. Most providers set a default timeout and a maximum timeout.

Keep in mind that each cloud provider may use different terminology and set unique limits on serverless functions, but the list above defines the basic concepts.

Serverless Architecture vs. Container Architecture

Both serverless and container architectures allow developers to deploy application code by abstracting away the host environment, but there are key differences between them. For example, developers who are using container architecture have to update and maintain each container they deploy, as well as its system settings and dependencies. In contrast, server maintenance in serverless architectures is handled entirely by the cloud provider. Additionally, serverless apps scale automatically, while scaling container architectures requires the use of an orchestration platform like Kubernetes.

Containers give developers control over the underlying operating system and runtime environment, making them suitable for applications that consistently get high traffic or as a first step in a cloud migration. Serverless functions, on the other hand, are better suited for trigger-based events such as payment processing.

Benefits and Challenges of Serverless Architecture

There has been a significant increase in serverless adoption in recent years, with nearly 40 percent of companies worldwide using it in some form. Small startups and global corporations alike are leveraging serverless architectures for the following reasons:

1. **Cost**

Cloud providers charge you on a per-invocation basis, so you're not paying for unused servers or virtual machines.

2. **Scalability**

Function instances are automatically created or removed in response to traffic variations, within the boundaries of concurrency limits.

3. **Productivity**

Engineers who use serverless can simply deploy their code without having to manage any servers, which helps accelerate delivery cycles and rapidly scale company operations.

There are also some challenges associated with serverless architectures:

1. **Loss of Control**

In serverless environments, you lack control over the software stack that your code runs on. If a hardware fault, data center outage, or other issue impacts one of your servers, you're dependent on a cloud provider to fix it.

2. **Security**

A cloud provider may run code from several of their customers on the same server at the same time. If the shared server isn't configured properly, your application data could be exposed.

3. **Performance Impact**

Cold starts are common in serverless environments, adding several seconds of latency to code execution when functions are invoked after a period of inactivity.

4. **Testing**

Developers can run unit tests on function code, but integration tests, which evaluate how frontend and backend components interact, are difficult to perform in a serverless environment.

5. **Vendor Lock-In**

Large cloud providers like AWS offer several services—such as databases, messaging queues, and APIs—that you can use in harmony to run serverless applications. Although it's possible to mix and match elements from different vendors, services from a single provider are designed to integrate most seamlessly.

Companies that want to minimize their go-to-market time and build scalable, lightweight applications can benefit greatly from serverless. But if your applications involve a large number of continuous, long-running processes, virtual machines or containers may be the better choice. In a hybrid infrastructure, developers may utilize containers or virtual machines to handle the bulk of requests but hand off certain short-running tasks, such as database storage, to serverless functions.

Serverless Architecture Use Cases

Serverless architecture is best used to perform short-lived tasks and manage workloads that experience infrequent or unpredictable traffic. The main use cases for serverless include:

1. **Trigger-based tasks**

Any user activity that triggers an event or a series of events is a good candidate for serverless architecture. For instance, a user signing up on your website may trigger a database change, which may, in turn, trigger a welcome email. The backend work can be handled through a chain of serverless functions.

2. **Building RESTful APIs**

You can leverage Amazon API Gateway with serverless functions to build RESTful APIs that scale with demand.

3. **Asynchronous processing**

Serverless functions can handle behind-the-scenes application tasks, such as rendering product information or transcoding videos after upload, without interrupting the flow of the application or adding user-facing latency.

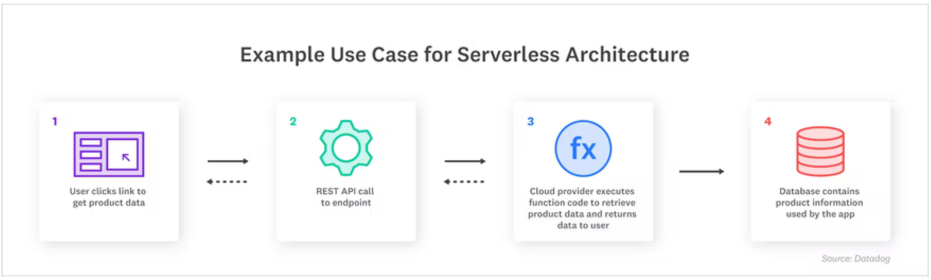
4. **Security checks**

When you spin up a new container, a function can be invoked to scan the instance for misconfigurations or vulnerabilities. Functions can also be used as a more secure option for SSH verification and two-factor authentication.

5. **Continuous Integration (CI) and Continuous Delivery (CD)**

Serverless architectures can automate many of the stages in your CI/CD pipelines. For example, code commits can trigger a function to create a build, and pull requests can trigger automated tests.

Most developers migrate to serverless in stages, slowly moving some parts of their application to serverless and leaving the rest on traditional servers. Serverless architectures are easily extensible, so you can always introduce more functions as opportunities arise.


Example Use Case for Serverless Architecture
1
User clicks link to get product data
2
REST API call to endpoint
3
fx
Cloud provider executes function code to retrieve product data and returns data to user
4
Database contains product information used by the app
Source: Datadog

Serverless architecture has numerous use cases, such as this trigger-based workflow for retrieving and displaying product information.

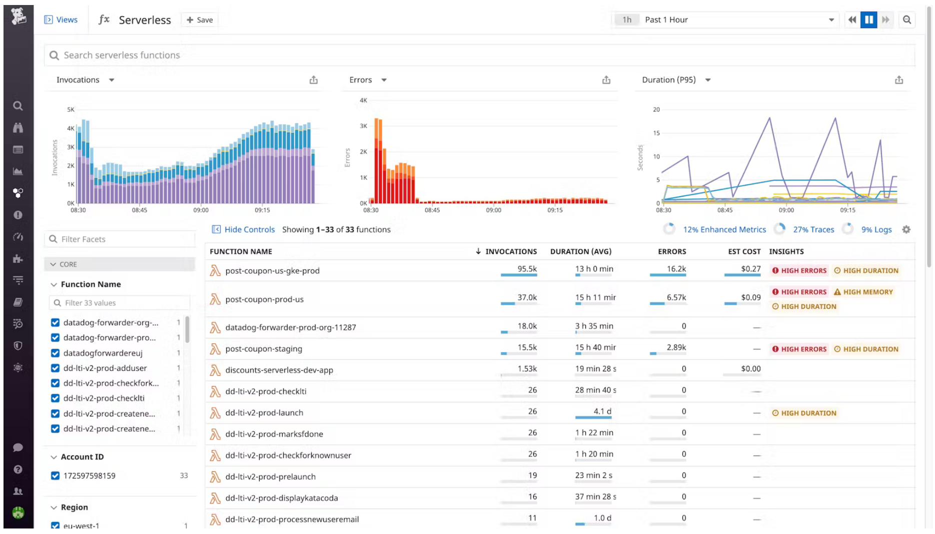
Tools That Support Serverless Architecture

The right tools can simplify the switch to serverless and ensure that your applications perform well for users.

A serverless deployment framework, such as Serverless Framework or Amazon's Serverless Application Model (SAM), interacts with the cloud provider's platform via an API and allows you to define your functions, triggers, and permissions. Some providers like AWS also offer serverless testing tools to locally test serverless applications prior to deployment. And serverless security tools scan your functions for vulnerabilities, with some even blocking code injections and unauthorized executables at runtime.

Once you've built your serverless application, you'll need to monitor its health and performance. Serverless functions typically travel through a complex web of microservices, and cold starts, misconfigurations, and other errors can occur at any node and cause ripple effects throughout your environment. To help you troubleshoot, it's critical to have real-time visibility into how each function is performing, both on its own and in communication with other functions and infrastructure components.

Datadog Serverless Monitoring offers end-to-end application monitoring whether your applications are completely serverless or run alongside containers and virtual machines. With Serverless Monitoring, you can observe the health and performance of your functions and other infrastructure components in real time, and collect metrics, traces, and logs from every invocation. Datadog supports multiple deployment frameworks and languages, so you can start monitoring your serverless architecture in minutes.



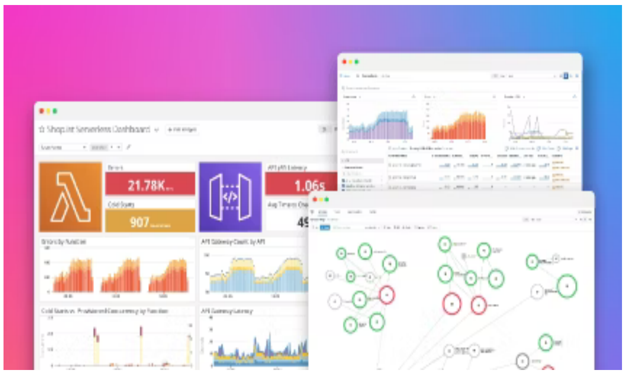
Hide Controls Showing 1–33 of 33 functions · 12% Enhanced Metrics · 27% Traces · 9% Logs

FUNCTION NAME	INVOCATIONS	DURATION (AVG)	ERRORS	EST COST	INSIGHTS
post-coupon-us-gke-prod	95.5k	13 h 0 min	16.2k	$0.27	HIGH ERRORS HIGH DURATION
post-coupon-prod-us	37.0k	15 h 11 mir	6.57k	$0.09	HIGH ERRORS HIGH MEMORY HIGH DURATION
datadog-forwarder-prod-org-11287	18.0k	3 h 35 min	0	—	
post-coupon-staging	15.5k	15 h 40 mir	2.89k	—	HIGH ERRORS HIGH DURATION
discounts-serverless-dev-app	1.53k	19 min 28 s	0	$0.00	
dd-lti-v2-prod-checklti	26	28 min 40 s	0	—	
dd-lti-v2-prod-launch	26	4.1 d	0	—	HIGH DURATION
dd-lti-v2-prod-marksfdone	26	1 h 22 min	0	—	
dd-lti-v2-prod-checkforknownuser	26	1 h 20 min	0	—	
dd-lti-v2-prod-prelaunch	19	23 min 2 s	0	—	
dd-lti-v2-prod-displaykatacoda	16	37 min 28 s	0	—	
dd-lti-v2-prod-processnewuseremail	11	1.0 d	0	—	

Datadog Serverless Monitoring makes it easy to track the health and performance of your functions.

Related Content

Learn about Datadog at your own pace with these on-demand resources.



PROD

Dat

End

end

Ser

Mo

Rea

BLOG

Key metrics for monitoring AWS Lambda

Read More

Get free unlimited monitoring for 14 days

START YOUR FREE TRIAL