IBM Instana

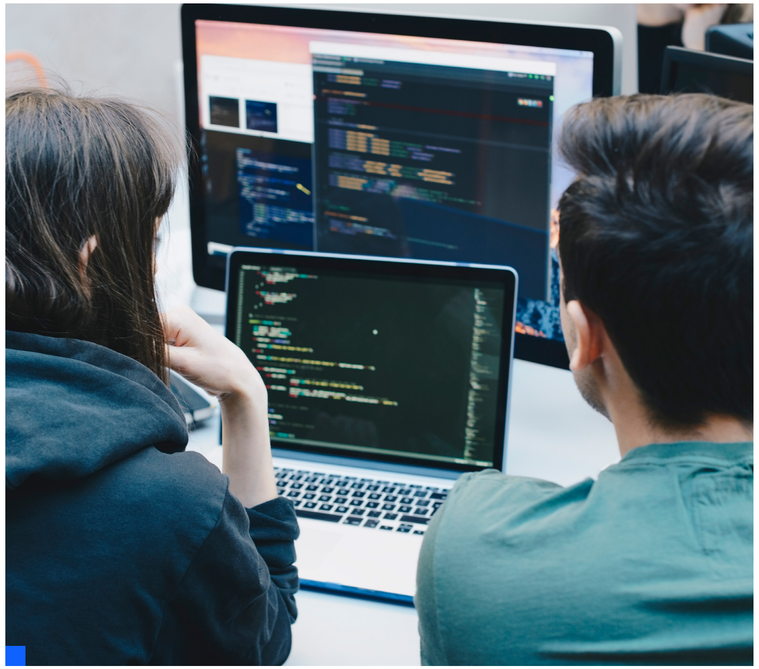
Monitoring serverless successfully

A comprehensive guide

Contents

01

Executive summary



Since the introduction of AWS Lambda in 2014, serverless computing has become one of the core building blocks of cloud-native infrastructure. By allowing organizations to run resource-intensive application code on demand in a cloud environment and pay for computing only when the code is running, serverless functions have unlocked new opportunities for optimizing application performance, availability and cost efficiency.

In each of these respects, serverless computing delivers critical benefits that aren't necessarily available from traditional physical servers or virtual machines (VMs), which can incur costs constantly—even when idle—and have limited resource allocations to handle intensive workloads.

At the same time, serverless technology has created new challenges for the teams tasked with monitoring and managing serverless functions. This is because serverless architectures are fundamentally different from conventional application deployment models, making it difficult to gain visibility into serverless environments and their functions.

This ebook offers a comprehensive look at the challenges of monitoring serverless functions, as well as the best practices for solving them. It analyzes the reasons that can make serverless monitoring so challenging and explains why traditional application performance monitoring (APM) tools can struggle when applied to serverless workloads. It identifies also a set of useful features to effectively monitor serverless environments while realizing the performance, cost and reliability advantages that a serverless architecture can offer.

02

Serverless monitoring challenges

While the core code within a serverless function may not fundamentally deviate from that found in a traditional application, the underlying architecture, hosting environment, and deployment patterns associated with serverless functions can exhibit significant differences. These disparities can give rise to numerous challenges when it comes to monitoring and maintaining visibility into serverless deployments.

In contrast, a conventional application operates in a consistent manner regardless of its hosting environment. For instance, an NGINX web server or WordPress instance hosted on Amazon Elastic Compute Cloud (Amazon EC2) virtual server functions and can be monitored in a similar manner as it would if it were hosted on the Microsoft Azure Virtual Machines service.

Vendor-specific serverless services
In serverless environments, the compatibility of workloads can become problematic. While all cloud-based serverless services offer basic functionality, each service is uniquely configured according to the vendor's specifications. For instance, AWS Lambda may not support the same programming languages as Azure Functions or Google Cloud Functions. Moreover, serverless services have distinct environment configurations that are typically non-modifiable by end users. The situation becomes even more diverse and inconsistent when incorporating on-prem serverless frameworks like OpenFaaS.

Consequently, attempting to monitor serverless functions solely at the service level can result in an approach that lacks portability and relies heavily on vendor-specific configurations. To avoid lock-in you would need to build a new monitoring process to support a transition to a serverless platform.

Therefore, monitoring serverless functions at the service level may present limitations, as the configuration levels depend on the environments and are vendor specific—meaning you'd need to build a new monitoring process from scratch to switch to a serverless platform.

Configuration levels are dependent on the environments and specific to each vendor.

Serverless functions come in many languages
Similarly, serverless functions can be written in a variety of different languages. And some serverless services support different languages than others. In some cases, serverless platforms may require functions written in one language to be "wrapped" in another to execute them, adding another layer of complexity.

Conventional APM methods, such as language-specific tracing and metrics collection, can be of limited use for gaining visibility into serverless performance and availability.

In other words, finding an easy or consistent way to monitor serverless functions by connecting through specific programming languages can be challenging.

Serverless functions are one piece of larger deployments

Serverless functions are often deployed as part of continuous delivery pipelines. It's rare to deploy a workload that's composed solely of serverless functions. In many cases, serverless functions comprise one part of a larger application. For example, most of the components of a web application might be hosted using traditional VMs and databases, while serverless functions are used to handle certain resource-intensive processes on demand, such as resizing images or performing optical character recognition (OCR).

For this reason, effectively monitoring serverless environments may require not only an understanding of what's happening within functions themselves, but also mapping that data to the performance of the larger application. Monitoring tools should be able to interpret the complex relationships and dependencies between each serverless function and the various nonserverless microservices that often power the rest of the workload.

Lack of control over serverless environments

Serverless environments eliminate the need for the teams who deploy serverless functions to set up or manage the servers that host them. Vendors deliver a prebuilt, preconfigured environment in which end users can quickly deploy and execute functions. These features make serverless environments so valuable.

However, end users can't access or modify the host environment, and this can pose a significant challenge for monitoring. Most serverless services do provide capabilities for forwarding log data to other cloud services, but teams often have limited ability to customize the way that data is generated or structured. Further, it's not always possible to deploy monitoring agents on host servers in a conventional way to collect and aggregate metrics.

Highly dynamic functions
Serverless environments change quickly. If monitoring tools must be manually mapped to a specific serverless function deployment, they need to be reconfigured manually whenever the functions are updated. This dependency means that monitoring can get in the way of continuous delivery—or worse—new versions of serverless functions might not be properly monitored because the tools aren't yet configured for the new deployment.

Too many functions
One of the final challenges in monitoring serverless functions is the sheer number of functions that serverless workloads entail. Although there's no minimum number of functions required to use a serverless service, teams that use serverless computing often deploy a dozen or more functions at the same time. They frequently introduce new functions and retire old ones on an ongoing basis.

Monitoring functions on this scale, using conventional, manual approaches, can prove difficult. It can require tremendous effort and time commitment on the part of administrators, and it undercuts the ability of teams to continue scaling up their deployments.

03

Why existing APM isn’t enough

A variety of traditional APM tools—designed before the advent of the cloud-native era—may support serverless monitoring to some extent, but they come up short in several respects:

- They can’t typically map and interpret the complex application architectures of which serverless functions are often a part.
- Designed for monitoring monolithic applications, they aren’t fully equipped to understand interservice dependencies or detect anomalies within highly dynamic environments.
- Usually vendor-specific, they can’t monitor serverless functions in a vendor-agnostic way, and they’re not portable from one cloud to another.

Ultimately, most existing APM solutions were designed for fundamentally different types of infrastructure. In general, they lack cloud-native monitoring functionality.

04

Selecting APM for serverless monitoring

Although most traditional APM solutions are generally inadequate for serverless monitoring, cloud-native APM tools generally offer the features required to handle the complexity, dynamism and scale of serverless computing services. Here is a summary of the key functionalities required in an APM solution to monitor serverless environments effectively.

Comprehensive tracing and analytics
There are two foundations to effective monitoring in cloud-native environments: tracing and analytics.

To effectively monitor serverless functions, APM tools must be able to perform tracing and analytics on individual parts of the application but also to perform them comprehensively, that is, across the entire application.

APM solutions must be able to trace and analyze individual application requests across every component of the application, including serverless functions and other services. At the same time, they must perform analytics on aggregate metrics collected from the application as a whole, as well as on collections of traces. Comprehensive tracing and analytics are the keys to providing visibility into all layers of a complex application and give teams the insight they need to address performance, availability and cost-optimization issues.

Dependency mapping
Because serverless functions depend on each other, as well as other application components that are external to the serverless environment, being able to map and interpret dependencies is critical.

APM tools that are designed only to monitor each part of the application individually, without understanding relationships between each part form a larger whole, are insufficient.

Auto discovery
If monitoring instrumentation must be configured manually each time a new serverless function or update is deployed, it's virtually impossible for monitoring tools to keep pace with continuous delivery chains at scale.

For that reason, serverless APM tools should be able to discover new deployments and updates automatically and then begin monitoring them without manual intervention by human engineers.

Instead, performance and cost optimization in a serverless workload usually requires the ability to determine how a problem with one serverless function impacts other functions or services within the application.

Cloud-agnostic serverless monitoring
As we mentioned before, serverless monitoring tools that work only with certain cloud services will depend on that vendor's specific services or configurations to monitor functions at the environment level.

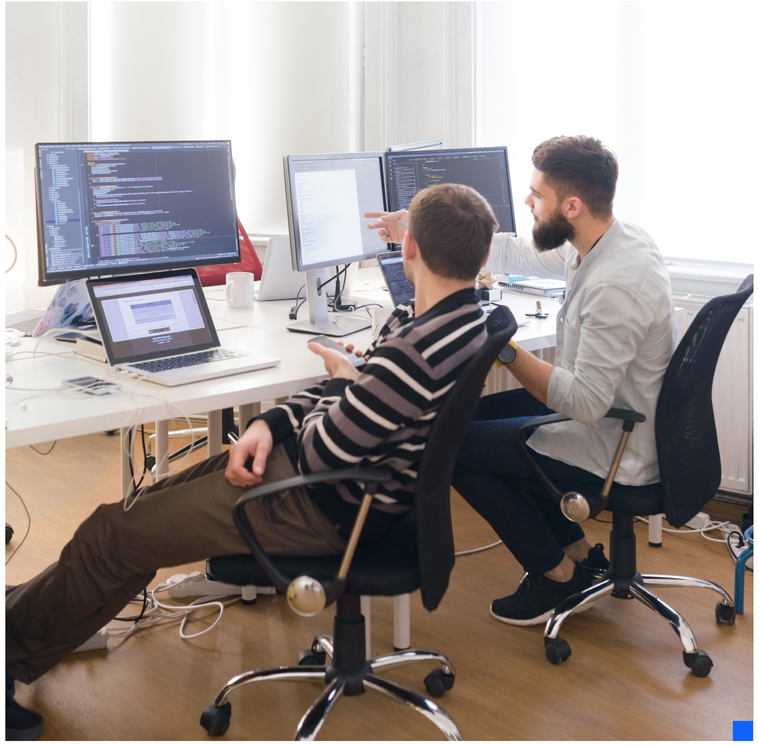
Choosing an APM tool that can monitor functions, regardless of which serverless service hosts them, can be a more flexible and portable approach. This approach requires the ability to perform tracing and analytics within functions themselves, rather than just connecting into the host environment.

Real-time and historical visualization
As a rule, successful serverless monitoring requires the ability to visualize analytics and traces clearly. Visualization helps monitoring teams make sense of and act on complex monitoring data and rapidly changing metrics.

To be most effective, APM tools must provide visualization for real-time and historical data. Real-time visualizations can give teams visibility into the application as it currently exists, while historical visualizations enable them to research an issue or gain crucial historical context when troubleshooting a problem.

05

Conclusion



Serverless monitoring requires a fundamentally different set of strategies and tooling than monitoring for conventional applications. Without an APM solution that can handle the unique challenges of serverless computing, organizations risk performance and availability problems that drive up costs and undercut the value of adopting serverless services.

Traditional APM tools aren’t designed to handle the complexity of serverless environments, or to keep pace with continuous delivery chains. But IBM Instana™, an APM platform designed from the start for the cloud-native age, does. Using data analytics and machine learning, IBM Instana maps the complex dependencies that link serverless functions to each other and to the rest of the application. This solution performs comprehensive tracing and analytics and provides rich visualizations that can help teams understand real-time and historical data. IBM Instana works in a cloud-agnostic way, allowing teams to monitor their serverless workloads across a range of serverless services. We invite you to sign up for a free IBM Instana trial.

Get your free trial →

06

About IBM Instana

IBM Instana™ provides an enterprise observability platform with automated application performance monitoring capabilities to businesses operating complex, modern, cloud-native applications no matter where they reside—on premises or in public and private clouds, including mobile devices or IBM zSystems™.

Control modern hybrid applications using IBM Instana for AI-powered discovery of deep contextual dependencies inside hybrid applications. IBM Instana also provides visibility into development pipelines to help enable closed-loop DevOps automation.

These capabilities provide actionable feedback needed for customers as they optimize application performance, enable innovation and mitigate risk, helping DevOps increase efficiency and add value to software delivery pipelines while meeting their service-level and business-level objectives.

Explore IBM Instana →

IBM Instana free trial →

IBM Corporation
New Orchard Road
Armonk, NY 10504

Produced in the United States of America
May 2023