dynatrace

Serverless computing at scale: Accelerating innovation with observability

What's inside

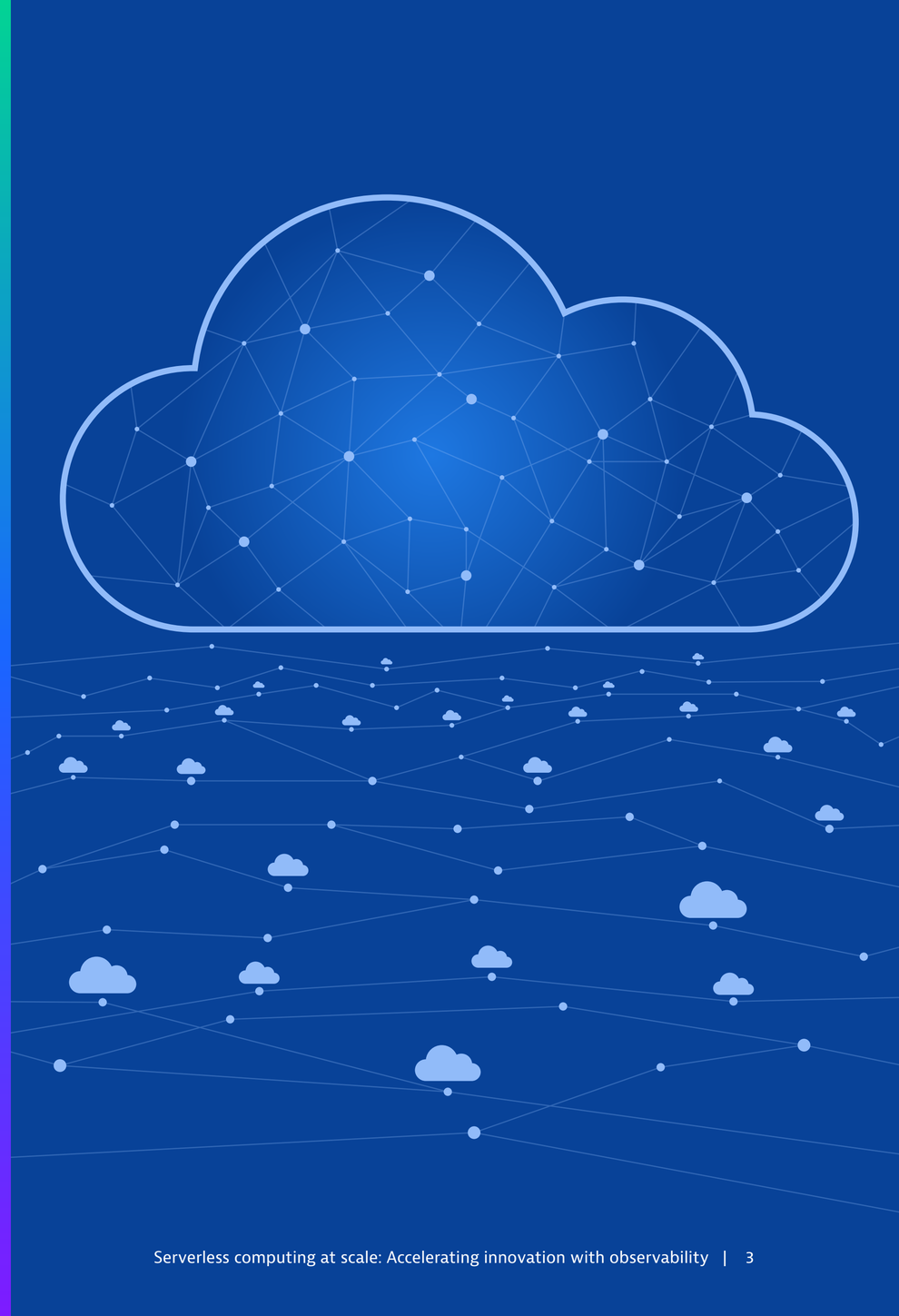
INTRODUCTION

Are organizations working harder, not smarter with serverless architecture?

Organizations are racing to digitally transform, with the goal of delivering innovative applications that grow the business. Serverless computing allows organizations to accelerate development and keep up with customer demand for online services by quickly and efficiently scaling their applications using an affordable pay-as-you-go pricing model. While the benefits are undeniable, serverless services can also introduce technical, administrative, and business challenges.

Microservices and cloud-native environments increase complexity, making it difficult to achieve observability across the entire application infrastructure. Services can span multiple cloud providers and connections, preventing IT teams from discerning where and when specific transactions happen. As a result, the organization struggles to identify the root causes of application performance issues that impact the user experience, making it much harder deliver on customer expectations.

Serverless has its upsides and downsides. But if you understand and plan for its advantages and shortcomings, you can design a serverless architecture that pays dividends. To build that foundation, we'll explore serverless technologies and the hyperscaler platforms, then consider some common use cases for when to use them. We'll also look at the benefits—and challenges—of serverless computing and offer some best practices for how to make your serverless architecture perform.



dynatrace

CHAPTER 1

What is serverless computing?

Serverless computing is a cloud-based, on-demand services execution framework in which users consume resources based on the applications and services they use. In a serverless computing model, cloud providers run the servers and their infrastructure, and dynamically manage machine resources according to demand.

Organizations turn to serverless computing to help their teams innovate faster and scale easier, and to reduce operational overhead. While the hyperscaler function-as-a-service (FaaS) platforms—such as AWS Lambda, Google Cloud Functions, and Azure Functions—are the most popular examples, the term serverless also includes container services, databases, and back-end services, among others.

How serverless computing works

Serverless computing streamlines operations and reduces costs associated with application development. In this paradigm, IT teams move their application hosting from local servers to cloud-based servers that providers manage on their behalf. Organizations can choose to purchase only the precise resources they need to build an application or service, then run it using an event-based, pay-per-use model. This cost-effective and flexible function-as-a-service (FaaS) arrangement doesn't require organizations to purchase and provision more server resources than they need in advance, and it can be spun up or down as needed.

DevOps teams appreciate serverless computing because it frees them to build applications and services without having to administer the underlying infrastructure or worry about capacity planning. With the freedom to focus on application design, deployment, and delivery instead of infrastructure, developers can apply their talent and expertise toward creating innovative applications and features. This flexibility enables organizations to transform faster and scale without worrying about the reliability challenges that would arise if they had to manage the entire application infrastructure on their own. appreciate serverless computing because it frees them to build applications and services without having to administer the underlying infrastructure or worry about capacity planning. With the freedom to focus on application design, deployment, and delivery instead of infrastructure, developers can apply their talent and expertise toward creating innovative applications and features. This flexibility enables organizations to transform faster and scale without worrying about the reliability challenges that would arise if they had to manage the entire application infrastructure on their own.

As a result, most organizations have multicloud environments. The flexibility and variety of services these platforms provide make them widely applicable to many use cases.

Meet the hyperscalers: AWS, Azure, and GCP

Three major cloud providers — or hyperscalers — offer serverless computing services that help organizations rapidly scale their applications: Amazon Web Services (AWS), Microsoft Azure, and Google Cloud Platform (GCP).

Amazon Web Services (AWS) offers 12 specific services that fall into three categories: compute, application integration, and data store. Its event-based function-as-a-service (Faas) computing platform, AWS Lambda, is one of its most popular services. Although it is possible to outsource almost any IT task using AWS services, organizations typically find AWS Lambda effective for underpinning critical application functions, improving data processing, boosting batch processing, and enhancing event ingestion., organizations typically find AWS particularly effective for underpinning critical application functions, improving data processing, boosting batch processing, and enhancing event ingestion.

Microsoft Azure offers a similar array of comprehensive serverless services. Its FaaS computing platform Azure Functions, for example, offers nearly the same capabilities as AWS Lambda. Organizations can use it to host databases, launch virtual servers, create websites or applications, run Kubernetes clusters, or even train machine learning models if they wish. Azure enables automatic scaling by reserving a base number of virtual machines and adding instances as needed during peak periods. Azure is especially well suited for smaller standalone apps and APIs, and less so for computationally demanding tasks that involve heavy CPU usage or infrequent, time-sensitive tasks.

Google Cloud, meanwhile, also offers its own comparable services infrastructure. Its FaaS platform, Google Cloud Functions (GCF), enables organizations to create and launch microservices. It pairs well with single-use functions that tie into other services. Like AWS and Azure, it also aims to help organizations simplify application development and innovate faster. Among other things, GCF is a good pick for use cases that involve creating backends, building integrations, completing processing tasks, or performing analysis. In general, GCF is best for situations that require enriching data and applying logic to dynamic conditions.

Because each cloud provider brings something unique to the table, organizations with complex requirements can and often purchase services from multiple providers to fulfill specific requirements. As a result, most organizations have multicloud environments. The flexibility and variety of services these platforms provide make them widely applicable to many use cases.

CHAPTER 2

Serverless computing use cases

To understand why serverless architecture can be such an advantage, it helps to consider some of the common ways organizations put these services to work.

Extend existing applications and infrastructure

Serverless computing offers organizations a strategic path to the cloud and the innovation it enables. Rather than diving directly into the deep end of the pool with disruptive application re-architecting, they can simply use services like Amazon Web Services (AWS) Lambda to extend their existing enterprise stacks into the cloud. For example, an organization might begin by transitioning its authentication or computational tasks to Lambda before wading in deeper with other AWS-managed services. Developers can easily access these services from within AWS Lambda functions, creating complex applications without needing to deploy a host or a container.

Organizations can also use serverless computing to store, analyze, and process application data. For example, Google Cloud Platform (GCP)'s Cloud Datastore service provides a highly scalable, cost-effective option compared with managing an in-house database such as Cassandra and MongoDB. Organizations also often use GCP for application analytics. Azure functions are a good pick for file processing and validation, ensuring that all files for a customer batch are fully prepared and that the structure of each file is properly validated.

Modularize processing architectures

Organizations also find serverless computing ideal for other use cases. For example, in the AWS family, they can use Lambda, API Gateway, and DynamoDB in tandem to create purpose-built, event-driven application back-ends that empower front-end functions. They can also leverage Lambda, Simple Storage Service (S3), Simple Queue Service (SQS), Simple Notification Service (SNS), and DynamoDB to develop and deploy general-purpose, event-driven parallel processing architectures that deliver real-time file processing, analysis, and output. AWS Lambda and SNS can be paired to enable better batch processing, making it possible to quickly download, upload, and process files while sending a timely notification to the IT team.

Build efficient single-page applications

AWS Lambda also serves as an efficient back-end for single-page applications (SPAs), which dynamically change their content using JavaScript, unlike regular web pages in which user interactions usually result in a complete page reload. Developers typically use front-end frameworks such as Angular, React.js, and Vue.js for their SPAs, relying on REST APIs to fetch their data using a method built into modern browsers (such as XHR or fetch). An AWS Lambda function located behind an API gateway can give developers just this kind of API endpoint to fetch data from other AWS services. Teams can even configure it to call back into an enterprise application to fetch data from a mainframe if needed. Meanwhile, services such as AWS and Cloudfront can host the static assets associated with an SPA, such as its JavaScript and HTML content.

Support mobile applications

Organizations can also use serverless computing to process events within their mobile applications. For example, in the case of a custom mobile application that generates events, organizations can create an AWS Lambda function to process those events as they are published. Developers, for instance, can set up a Lambda function to process the clicks that take place within the custom mobile application. Azure functions, meanwhile, can be used to implement an API for a mobile application and build a cross-platform client spanning Android, iOS, and Windows.

As organizations use serverless computing for an increasing range of use cases, they need to ensure proper monitoring across the full stack. Not only is observability important for hybrid environments, it's also essential for multicloud scenarios involving multiple cloud providers. An end-to-end observability solution can give organizations the awareness and insight they need across their entire application infrastructure. With visibility into both on-premises and cloud resources, organizations can access all the benefits serverless computing offers without sacrificing performance or operational agility.

CHAPTER 3

The benefits of serverless frameworks

Serverless is more than just a trend. This cloud-based model offers real benefits to organizations and their customers. Organizations can measure the benefits of this architecture approach across two key areas: business and technical.

dynatrace

Business benefits of making the shift to serverless

One industry study found that 40% of organizations have adopted serverless technologies such as AWS Lambda in order to leverage key operational benefits. These benefits include the following:

Improved customer satisfaction

Serverless computing can help to streamline resource-intensive processes such as user verification and authentication, in turn reducing the time required to confirm user identities. The result? Improved customer satisfaction when users try to log into their accounts or make purchases since they don't need to wait for local servers to receive, process, and return authentication data.

Increased revenue and conversion

Using serverless solutions also simplifies the process of building APIs and connecting edge services in the cloud to deliver a seamless experience to customers — such as when they use a mobile app or log into the e-commerce site. The result is increased engagement that drives improved conversion and return on investment (ROI) because customers can rely on the applications to perform consistently over time.

Accelerated application release cycles

By automating core processes and providing a cloud-based repository of all application changes, serverless makes it possible to shorten the time between app design, development, and release. This helps to get the apps to market faster and with fewer potential problems. Also, if changes are required, less time is lost between revision and redeployment.

Improved cost management

Compute and resource demand aren't fixed. Under a traditional provisioning model, companies purchase more resources than they need in case of demand spikes. In a serverless environment, businesses only pay for what they use. The result? Improved cost management since spend is tied to actual use, rather than anticipatory demand.

The technical benefits of serverless solutions

Serverless architecture also offers advantages for technical teams, such as the following:

Scalability

Using a serverless approach makes it possible to scale cloud compute power efficiently. When introduced to the public cloud, serverless frameworks become both flexible and autonomous because the scalability of resources and applications is no longer limited by underlying infrastructure.

Reliability

Serverless architecture provides natural redundancy, thus improving overall reliability. Consider a sudden on-site or application failure — the agile nature of resource allocation in a serverless model means that both applications and core services can be back up and running in a fraction of the time required for traditional infrastructure.

Performance

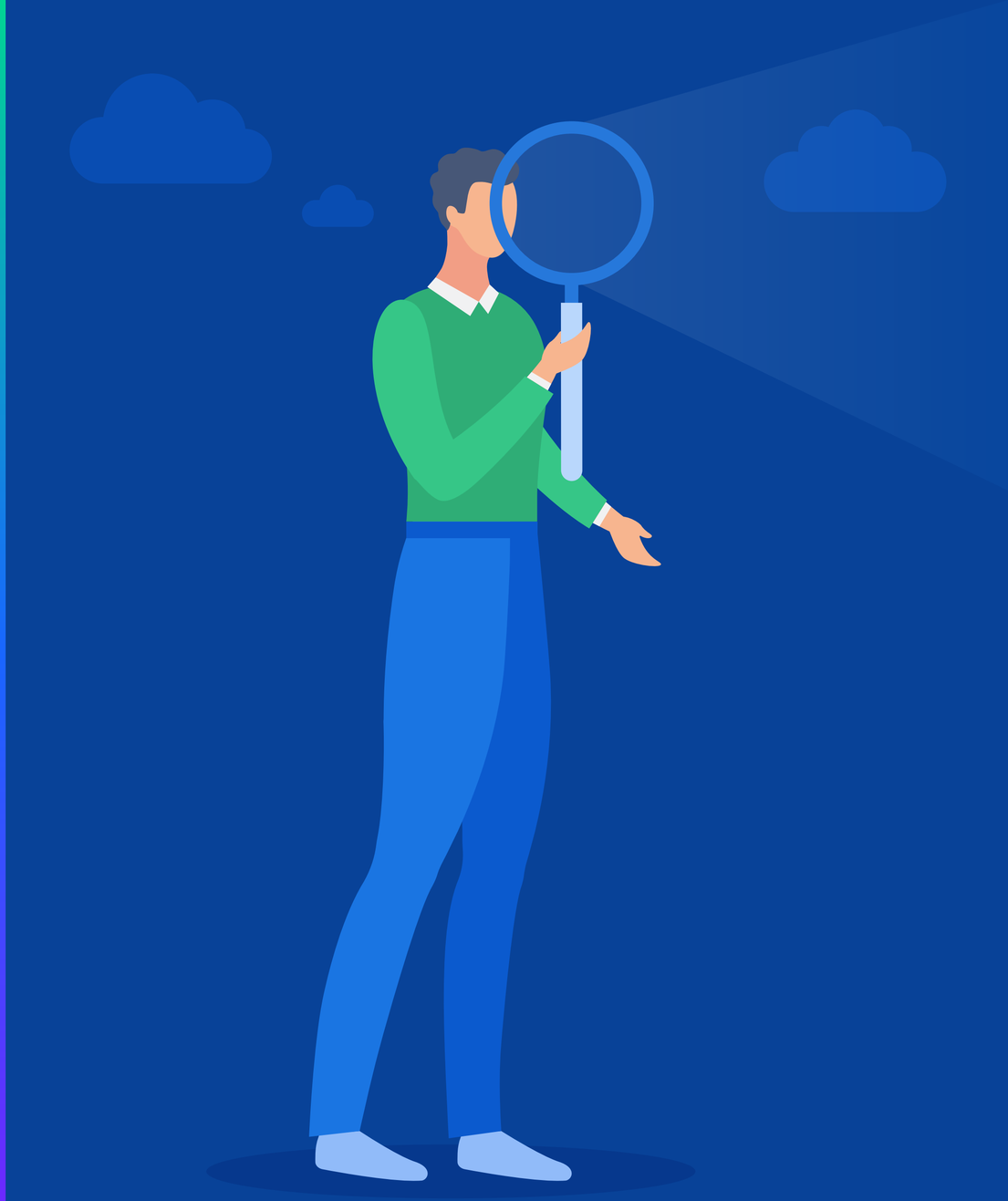
By allowing organizations to purchase resources under an event-based, pay-per-use model, serverless solutions make it possible for teams to pick and choose services that deliver optimal outcomes. For example, organizations might choose to purchase only the central processing unit (CPU) cycles and memory needed to support current application performance, and then scale up as needed to meet increasing demand.

Despite the benefits of serverless computing and how this approach accelerates innovation at scale, it can also lead to observability problems.

CHAPTER 4

The observability challenges of serverless computing

Serverless architectures can improve availability and performance, but they can bring complexity that presents observability challenges. Understanding—and avoiding—the common observability obstacles that come with serverless are crucial to the success of making the move.



dynatrace

What is observability?

Observability means that a system provides telemetry data that enables IT professionals to examine specific processes or operations in order to observe operations in real-time. It makes it possible for teams to see what's happening across a serverless environment at any given moment. Observability allows teams to quickly pinpoint potential problems with services as they arise rather than wait for downstream results—such as a system outage—to indicate the presence of problems. It's no surprise, then, that 93% of IT leaders say that observability is now foundational for ongoing business success.

Why is observability challenging with serverless?

While observable systems are a key component of serverless success, achieving reliable observability data is often problematic in practice because of three common challenges:

1. **Dynamic operating environments:** Serverless solutions streamline operations across multiple platforms, technologies, and infrastructure deployments. The dynamic nature of serverless environments creates observability challenges because constantly changing services make it difficult to follow distributed traces—the start-to-finish records of all events that occur along the path of a given request. As a result, organizations often rely on manual efforts to account for observability gaps, which creates potential blind spots.

2. **Increased volume, variety, and velocity of services:** Today, the opportunities to use serverless technologies are seemingly endless. For example, open source software, applications, infrastructure, microservices, and so on. But as the volume and variety of deployed services continue to grow, the sheer velocity of information created also ramps up. This often creates an observability bottleneck that sees teams inundated with a host of information about the states, locations, and dependencies of services—making them miss the context necessary to develop a holistic view of operations.

3. **Multicloud management:** As organizations increasingly adopt multicloud technologies to fulfill specific tasks, these environments also present challenges for observability. While each cloud environment—such as AWS, Azure, or Google—has its own set of monitoring and observability tools, these tools often support just the services from the cloud hyperscaler itself along with any services that immediately touch the platform. In practice, this means that while teams have visibility across the services and tools leveraged by a specific cloud provider, observability vanishes in the gaps between clouds—thus, making it difficult to track operations end-to-end.

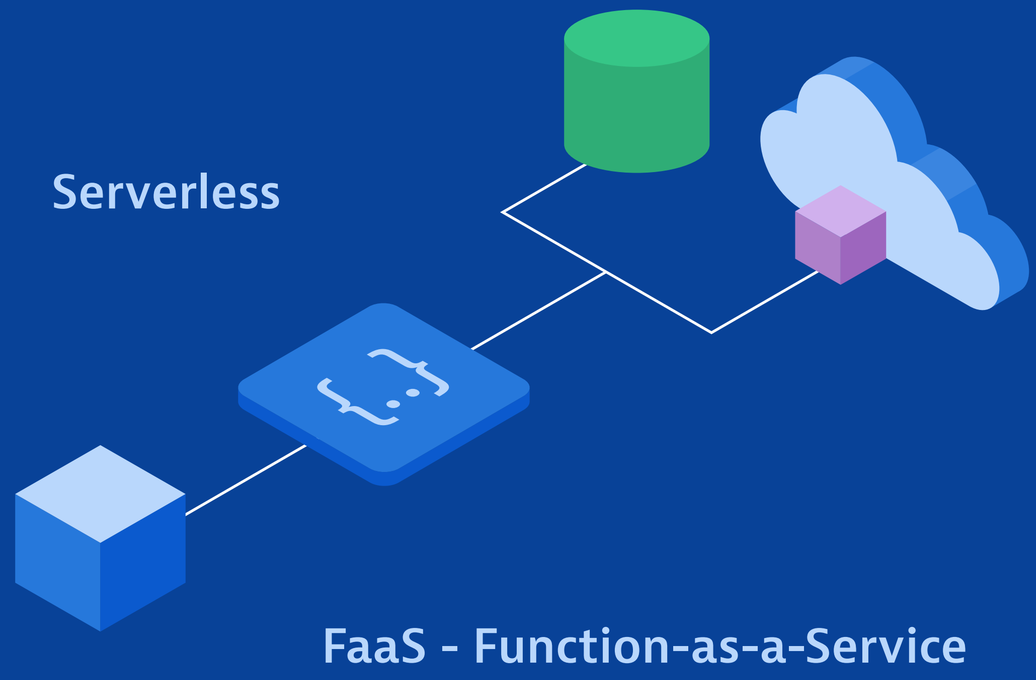
Put simply? While serverless solutions offer substantive benefits, organizations must effectively address observability challenges to make the best use of this cloud-native approach.

CHAPTER 5

Best practices for optimizing multicloud serverless architectures

Serverless observability and monitoring can be challenging due to the highly distributed nature of its architecture. This issue is amplified when organizations use more than one cloud provider. Moreover, applications that leverage serverless technology typically have complex service call chains with limited visibility into the underlying infrastructure.

While all serverless platforms are unique, the following observability best practices are essential for the success of any serverless implementation.



1 Design serverless applications for observability

Serverless application development significantly differs from traditional development in that system health monitoring is the responsibility of the FaaS cloud provider. This includes the physical infrastructure, virtualization, operating system layers, and platform runtime of cloud providers such as AWS , Azure, or Google Cloud Platform.

When designing an application, consider the telemetry you need to determine application health in relation to the state of the underlying cloud infrastructure. But monitoring the success of your business objectives is outside the scope of the FaaS provider. Therefore, be sure to build in the relevant telemetry you need to support your business objectives through end-to-end traces and spans.

2 Centralize observability

Most serverless architectures span multiple cloud environments, technologies, and services, which makes it important to take a centralized platform approach to observability.

The benefits of centralizing observability into a single view—one that spans hybrid-cloud environments, distributed microservices, applications, open-source software, and Kubernetes containers—are two-fold. First, a centralized approach gives teams a single source of truth for tracking all performance and system health issues. Second, centralized observability breaks down data silos between teams, providing developers and business analysts crucial insight into serverless efficiency and its impact on user outcomes.

3 Monitor cold starts, latency, and invocation errors

For many serverless platforms, memory-heavy multi-function applications can be resource-intensive, especially when they aggressively scale. Infrequent, time-sensitive tasks sometimes result in slower response times because of cold starts.

A cold start is when a request comes in for a serverless function but an idle FaaS microservices container is not available— therefore, the system needs to spin up new container. This can cause an increase in latency, such as a delay before the requested information loads. If a request is rejected before the function receives it, it causes an invocation error.

To prevent delays in response time, it's important to monitor each individual invocation to detect when a function is cold started. Then you can design strategies to prevent it, such as warming up the functions or configuring provisioned concurrency.

4 Automate

Because serverless is dynamic, another best practice is to use automation to instrument and monitor serverless functions. Automated monitoring and observability of serverless functions optimizes performance while providing real-time insights on impacts to the serverless environment.

5 Establish distributed tracing

End-to-end distributed tracing that follows the chain of calls between the event trigger, serverless functions (i.e., Lambda), and cloud services is critical for serverless architectures. It is especially important that distributed tracing be able to span multiple cloud environments, on-premises infrastructure, open source software, container environments, and their payloads to get full end-to-end context. Distributed tracing helps to identify and remediate the root cause of an issue before it impacts end users.

6 Incorporate performance metrics

On-demand Lambda functions exist only for the duration of the request, with logs spread across multiple resources. To ensure overall system performance, it is essential to collect and monitor these metrics from end to end. Essential performance metrics to monitor on both the platform and the serverless applications include the following:

- Operational metrics such as aggregate error and usage count
- Load-related metrics such as CPU, RAM, and network bandwidth
- Business metrics to monitor application health and effect on end users

7 Implement AI

Serverless functions generate massive volumes of telemetry data that make full visibility into the serverless stack impossible without the help of AI. AI-powered multicloud monitoring automatically provides real-time context for each serverless function in relation to other services across the full stack — regardless of whether it is called via HTTPS or an SDK. This cloud-native AI intelligence provides automatic, precise answers to serverless application anomalies before they impact customers.

8 Extend observability to end users

Full serverless observability requires digital experience monitoring (DEM) to provide data and insights on how end- users are affected by serverless functions along their digital journey. DEM can detect technical failures before they affect customers and help prioritize issues that will have the most impact on critical business KPIs.

By integrating DEM into a platform-based approach, teams can combine application performance data, real-user behavior, synthetic monitoring, and deep experience insights to pinpoint issues affecting users in real-time, ensuring serverless applications are available, functional, fast, and efficient across all digital channels.

CONCLUSION

Taming serverless complexity with automatic and intelligent observability

For organizations seeking a competitive edge and ways to operate more efficiently, serverless architecture helps achieve these goals. A serverless application infrastructure can help IT teams improve customer experience, boost revenue, and accelerate software development.

dynatrace

Because serverless computing outsources infrastructure to a public cloud provider or hosting provider, organizations can sidestep the cost and time associated with managing infrastructure on-premises, while also gaining the flexibility to scale rapidly and efficiently.

Serverless architecture reduces the friction that traditional IT infrastructure management requires. With serverless infrastructure, IT can streamline resource-intensive processes such as user authentication and verification with fewer delays and steps. Serverless architecture also simplifies the process of application interfaces and connecting edge services to deliver seamless customer experience. Further, serverless architecture accelerates software development by shortening the time between application development and release. By reducing workflow friction and time spent on manual tasks, organizations are free to focus on higher-level tasks. Ultimately, these reductions enable organizations to accelerate their pace of product innovation.

To succeed with serverless architecture, however, IT teams need to enlist best practices that focus on solving the observability issues in multicloud environments.

First, they need to think about how to architect applications in a serverless environment. They need to be able to generate telemetry data to determine application health in the context of the wider cloud infrastructure. Second, organizations need a unified observability platform that can cross all data silos and provide a single source of truth for IT teams. Third, a serverless architecture requires a centralized platform with cloud-native AI intelligence at its heart that automatically delivers precise answers about system issues and application anomalies before they undermine user experience.

Armed with this kind of intelligence, organizations can spend less time troubleshooting and earn the crucial customer trust that is essential for long-term business success.

Automatic and intelligent observability for hybrid multiclouds

We hope this ebook has inspired you to take the next step in your digital journey. Dynatrace is committed to providing enterprises the data and intelligence they need to be successful with their enterprise cloud and digital transformation initiatives, no matter how complex.

Learn more

If you are ready to learn more, please visit www.dynatrace.com/platform for assets, resources, and a **free 15-day trial**.

dynatrace

Dynatrace (NYSE: DT) exists to make the world's software work perfectly. Our unified software intelligence platform combines broad and deep observability and continuous runtime application security with the most advanced AIOps to provide answers and intelligent automation from data at enormous scale. This enables innovators to modernize and automate cloud operations, deliver software faster and more securely, and ensure flawless digital experiences. That's why the world's largest organizations trust the Dynatrace® platform to accelerate digital transformation.

Curious to see how you can simplify your cloud and maximize the impact of your digital teams? Let us show you. Sign up for a free 15-day Dynatrace trial.

blog @dynatrace