IBM Instana

Gaining observability in cloud-native applications

Contents

01

Introduction

The widespread adoption of diverse cloud-native technologies has resulted in significant changes in application development, delivery and operations, leading to a new competitive landscape. Faster organizations are capitalizing on this shift to outmaneuver their larger, slower counterparts.

The changes to the application tech stack were swift. Cloud environments, public and private, expanded, virtual servers evolved into containers, and Kubernetes became the orchestration standard. Container adoption exploded and evolved into serverless workloads in serverless environments while Dev teams became more agile. As software and infrastructure blurred, two competing constants emerged:

- Application performance monitoring (APM) became an important contributor to business success.
- It was difficult to gain visibility into modern applications using traditional APM methodologies.

Thus, the concept of observability—in many forms—emerged as organizations battled with the need to accelerate development and delivery while maintaining the visibility IT operations teams expected from APM tools. But observability has its challenges, including many of the same faced by traditional monitoring tools. Speedy, flexible application delivery organizations such as Dev, Ops, DevOps, and SREs, want to use the entire set of cloud-native technologies. To do that, they need a comprehensive approach to observability that takes advantage of the best practices of developer-driven visibility and modern APM solutions.

02

Cloud-based versus cloud-native technologies

Cloud technology has been a real watershed in terms of application development. Cloud and multicloud environments provide more power and redundancy, but this necessitates the modernization of applications. While cloud-based applications have been retrofitted for operation in the cloud, cloud-native applications were developed specifically for cloud deployment. However, whose cloud are we referring to? As companies opt for multicloud environments, the complexity increases. Hybrid clouds, which comprise a mixture of private and public clouds, further compound this complexity. When we add a layer of on-premises hosting to the mix, modern technology stacks can exert additional pressure on development teams, making it challenging to ensure timely delivery of applications.

Monitoring the performance of cloud-native applications can also prove challenging because it can be difficult for traditional APM and observability tools to see inside cloud-native elements such as containers as easily as they see into other application stacks. In contrast, cloud-native observability platforms tend to track all the metrics, event traces and logs across the full stack to provide broader visibility.

↓

Cloud-based applications have been retrofitted for operation in the cloud.

↓

Cloud-native applications were developed specifically for cloud.

Observability in cloud-native environments

The technologies that make up modern cloud-native technology stacks present some specific obstacles for traditional monitoring tools:

Lack of backward compatibility
Containers are purposely built to contain only required services to keep them lightweight, which means older instrumentation engines may not have access to required services needed to work.

Orchestration information correlation
Kubernetes blurs the lines between software and infrastructure to the point that existing tools may have to use multiple agents just to extract basic data, which would eliminate the ability to directly correlate information.

Serverless elimination
Serverless architectures take the service elimination ideal to an even greater level by taking the servers from local control to the cloud platform, leading to the same service availability issues.

Accelerating development and deployment rates
As changes occur in the application environment due to the ephemeral nature of containers, it can be difficult for traditional monitoring tools to reach their accustomed level of visibility.

As traditional APM tools struggled to meet requirements in cloud-native environments, alternatives for application observability emerged thanks to other factors:

- Developers became more involved in operating production workloads and assuring application performance.
- Cloud-native tech stacks enabled accelerated development and application pipelines, meaning more pieces could be operating with visibility gaps from existing tools.

Observability provides external indicators of the health and status of applications inside cloud-native elements such as containers, microservices, orchestration tools such as Kubernetes, and serverless environments from a performance perspective.

Observability challenges

From “black boxes”—the hidden internal workings of a product or program—to massively short-lived deployments, the cloud-native stack creates numerous challenges for observability and APM tools:

Traditional monitoring tools can’t handle distributed microservice environments
Environments for container-based applications tend to be dynamic, distributed microservices, deployed across clusters of servers that are also dynamic. One complete application could cross different kinds of infrastructures, platforms, languages and technologies.

Traditional infrastructure and application monitoring tools were designed for monolithic applications mapped to static individual servers, and they tend to focus on providing visibility at the language level. These traditional monitoring approaches can break down in the face of microservices distributed across a large cluster of servers composed of multiple languages, middleware components and database systems.

Containers host a wide variety of workloads
Containers can host various types of workloads, ranging from monolithic applications like Apache HTTP servers to Java virtual machines and databases, creating complex monitoring challenges. The outsourced persistent data storage on permanent servers in a hybrid environment further complicates matters. With such workload variability in containerized environments, monitoring tools must support a diverse range of technologies, architectures, and configurations, making it a daunting task.

Traditional monitoring tools can be challenging to configure for containers, as every container is different, and it's difficult to determine the thresholds to set in advance for each one. Additionally, monitoring tools must be automated to handle unpredictable workload types as manual configuration and enforcement of monitoring policies are not feasible.

Containers require elevated monitoring
Container platforms typically include only basic monitoring functionality, but the stats tool is insufficient for monitoring large-scale production environments. This separates containers from other application infrastructure technologies.

Similarly, operating systems provide significant data about system performance, but they don’t include application-level data. Even advanced virtual machine platforms such as VMware, which include relatively sophisticated monitoring tools, fail to provide application layer data or distributed tracing.

Kubernetes is not a performance monitoring tool
Container orchestrators such as Kubernetes are great for organizing, provisioning and managing the deployment of their production container environments and applications.

But orchestrators are not designed for monitoring, and although they have tremendous focus on container and host resources, orchestration management includes no aspect of user experience or application performance.

Furthermore, container orchestrators can only manage things that they are aware of—which mean the infrastructure and services running in containers. In a hybrid environment, container orchestrators, by definition, don’t monitor resource usage of any other resources. That can leave a large part of your environment at risk for performance problems.

05

Critical characteristics of observability for cloud-native environments

Visibility is the heartbeat of cloud-native observability platforms. The ability to monitor containers from within is a critical feature for dynamic microservices-based architectures in which tracing and dependency maps can be a challenge.

Even with the ephemeral and sometimes incomplete nature of containers, cloud-native observability provides actionable insights to turn your teams from passive observers into active participants who can resolve and even prevent issues. Agents knowledgeable in containerized microservices can map the resources in an IT architecture, even when those resources are barely related and constantly changing.

Other critical characteristics include:

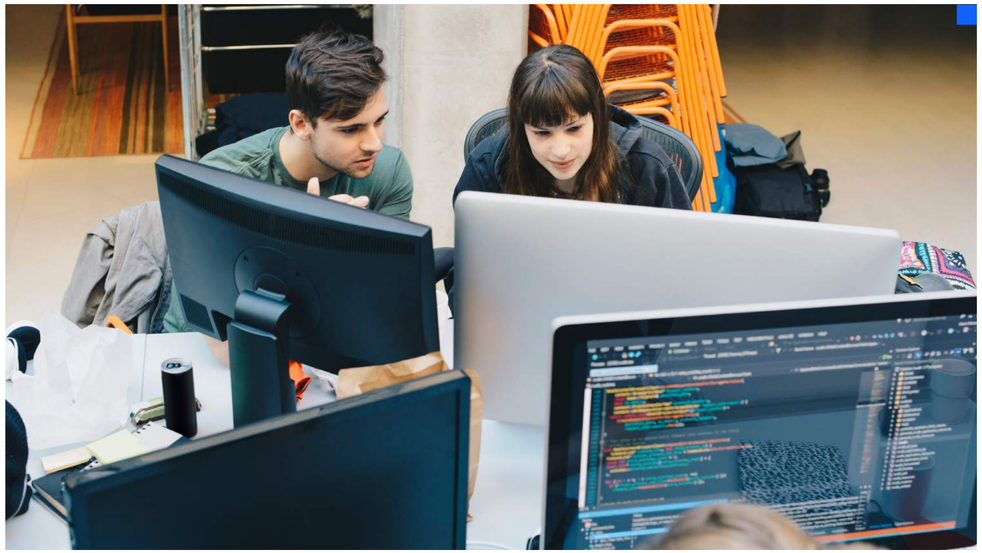
All the data
DevOps must have access to complete analysis to understand exactly how, when and where problems exist. The only way to achieve this is to capture a distributed trace of every request—no gaps, no sampling and no proxies. Additionally, every piece of the infrastructure, platform and line of code deployed into the containerized environment must support tracing.

Cloud-native deployment
Cloud-native observability tooling integrates seamlessly into cloud-native application environments, fully automated deployments and instrumentation processes.

Root cause analysis
Regardless of how complex your environment is, DevOps teams must identify and resolve performance issues as quickly as possible. An observability solution must be able to automatically point you to where and why distributed applications break down.

06

The risks in not adopting observability for cloud-native environments



For some organizations, deploying observability for cloud-native applications may not be a priority, perhaps because their APM solution isn't presenting an alarming number of errors or because their teams would need to learn a new platform.

But if your traditional monitoring solution is missing errors, transaction latency and opportunities for optimization, you might not be aware that you need an enterprise observability platform. Delay comes with risk:

- An undetected error could result in customer-facing service degradation or an outage.
- Delays from time of incident detection to the time of discovery by a human could be the difference between incident prevention and major incident management.
- Your monitoring tool could find an issue but not provide enough contextual information for you to react.

All these scenarios represent potential damage to your reputation, loss of revenue due to downtime, and affect your customer experience.

07

Benefits of observability for cloud-native applications

Of course, the primary benefit of a more comprehensive monitoring solution is to discover issues and prevent incidents. With observability, transparent and observable systems allow engineers and SREs to free up the time–previously required for continuous monitoring–and focus on higher value-added tasks like producing new product features for customers.

Faster continuous integration/continuous deployment (CI/CD) processes, less wasted work, fewer errors and happy customers save time and money. But IBM Instana™ offers much more:

Reduce mean time to repair (MTTR)
Event correlation can be the difference between triaging an incident for hours and resolving an impending issue before it can even happen.

Shift left
The earlier your engineers and developers can become aware of an issue, the more proactive and less reactive they can be, and the less impact every issue will have.

Reduce CI/CD impact
Observability makes it easier to track the success of a canary release or a blue/green deployment in progress. Once unsuccessful code is released into production, damage is done.

Healthier systems
The natural output of an observability tool in a cloud-native environment with DevOps means systems with fewer errors and more efficient processes.

Happier customers
As a customer, how long would you continue to pay for an application or service that has errors, responds slowly, experiences downtime and threatens your business?

08

Introducing enterprise observability

Although observability can deliver the signals you need to track application performance, getting to a truly comprehensive understanding of your entire system is better served by enterprise observability.

With enterprise observability, you can do more than just monitor individual systems. You can contextualize data about those systems and correlate interactions between applications and systems across your entire IT environment. The foundations of enterprise observability are automation, context, intelligent action, remediation and ease of use.

More specifically, enterprise observability is founded on several key practices and principles:

- **Systematic optimization**
  Enterprise observability focuses not on managing the health and performance of individual applications or systems but on optimizing the entire IT environment. That means mapping and contextualizing all resources within the architecture, even if they are constantly changing or loosely coupled.

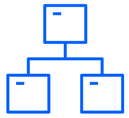
- **Complete contextualization**
  Every unit of observability data must be delivered with complete context. Teams cannot rely on sampling and guessing; they need end-to-end tracing and contextualization.

- **Cloud-native deployment**
  Enterprise observability tooling must be able to integrate seamlessly into the cloud-native application environments that it supports. The deployment and instrumentation processes are fully automated.

- **Comprehensive support for data ingestion**
  Modern enterprise application environments expose data in various ways: as standard output or conventional logs or through open-source monitoring APIs. Enterprise observability tools must support all data formats to ingest and contextualize every data source.

- **Observability across the pipeline**
  Teams must be able to monitor and contextualize application behavior starting at the beginning of the CI/CD pipeline and continuing through to deployment. You shouldn't have to wait until a new application release is in production to be able to understand how it interacts with other systems and optimize its behavior.

Automation
When new code is deployed or system changes are made, enterprise observability automatically and continuously discovers changes and provides immediate feedback.



Context
Enterprise observability explains how every application component and service interrelates with every other component and service to optimize performance and availability of resources.

Intelligent action
When changes occur, enterprise observability proactively provides deep analysis with context and suggests steps for system optimizations.

Ease of use
As organizations roll out agile development processes, increase the frequency of their application updates and fill their CI/CD pipelines, more stakeholders require actionable information.

09

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 with IBM Instana's 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.

For more information
To learn more, contact your IBM Business Partner

BATEC

973 973 36 565 551 / 973 3805 9706 / 973 17740400 |
batecibm@batecict.com / treesa@batecict.com

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 →