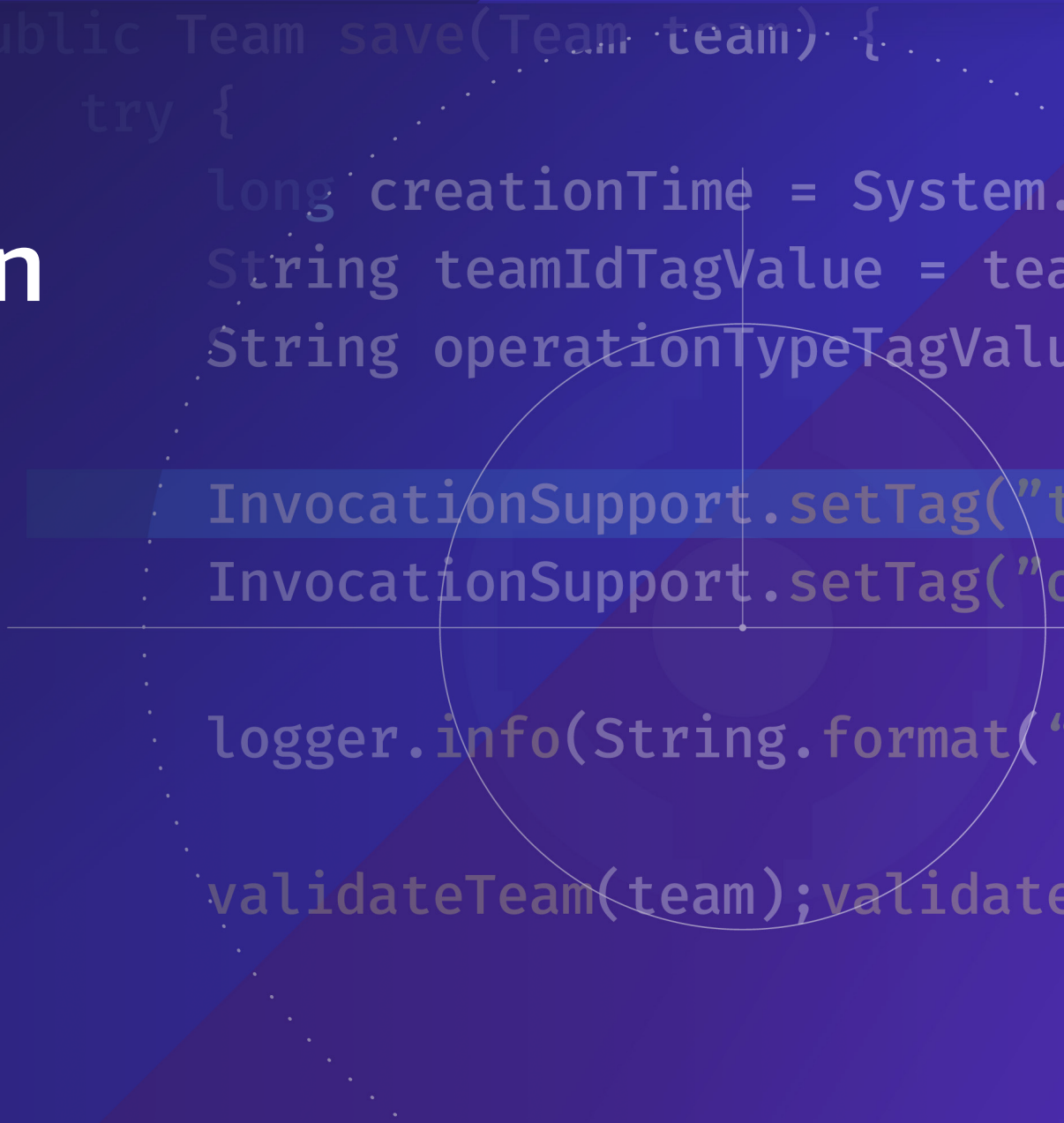
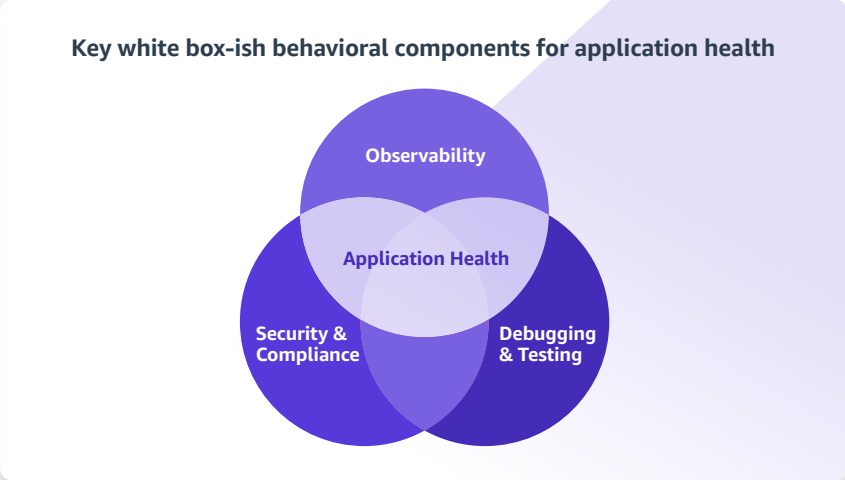
aws | THUNDRA™

EBOOK

Mastering Observability on the Cloud: New Requirements Require a New Approach

Discover key capabilities that enable your teams to do more with serverless applications





Interested in learning more about serverless applications? Explore these resources:

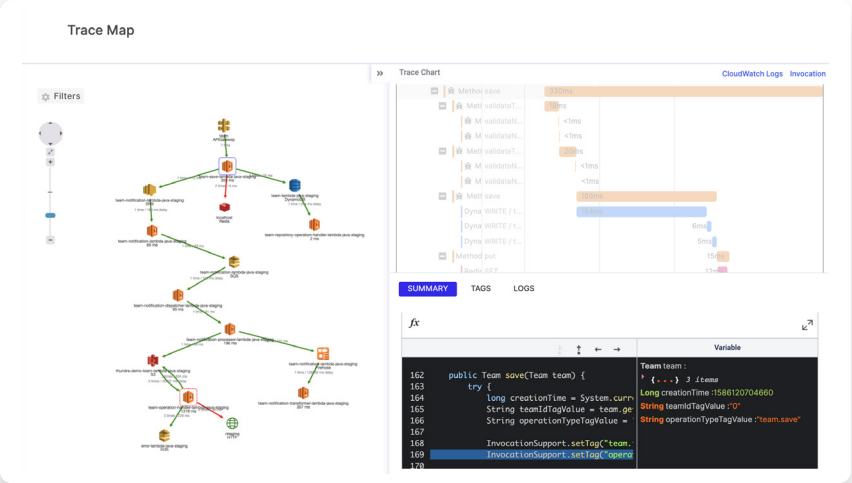
- Serverless is Taking Off: Here's Why It's Worth Hopping On
- Understanding Serverless Application Behavior with Thundra
- Putting an End to the Misery of Serverless Application Debugging With Thundra
- How to Secure Serverless Applications

Focus on the economic unit of value of IT investments: your applications

For enterprises, microservices applications using serverless functions and services present a new way to build software that is cost effective and improves efficiency and performance. Because they run on cloud-native environments, this removes much of the burden from development teams; instead of worrying about underlying infrastructure, they can focus on building new features and innovating.

Companies face new challenges, though, as they modernize. In this eBook, we will discuss why you need a new approach for modern applications. We'll also address three key questions that teams face to help you better manage your serverless applications:

- **How do I understand** the behavior of my application end-to-end from an observability perspective?
- **How do I troubleshoot** and debug my applications as productively and cost effectively as possible?
- **How do I ensure** my applications meet company requirements from a security, compliance, and architectural best practices perspective?



Thundra End-to-End Tracing

LEARN MORE **about why end-to-end tracing is important for serverless.**

How do I understand the behavior of my application end-to-end from an observability perspective?

The ability for developers to understand application behavior end-to-end allows them to rapidly troubleshoot and debug. It also enables DevOps teams to quickly identify symptoms that may be caused by underlying infrastructure issues. This understanding is key to building performant applications designed for resiliency at scale.

Microservices applications, which are complex and composed of loosely decoupled distributed services, makes this challenging. **To achieve a true understanding, developers must manage the behavior of:**

- **Each service**
- **Synchronous web-based communications (APIs/HTTP)**
- **Asynchronous event-based communications**
- **The compute resource at runtime**
- **The executing code**

Understanding symptoms is not the same as understanding the root cause of an issue. Being able to identify the root cause of an issue allows you to understand at a holistic level and trace what's happening across multiple distributed services.

Thundra helps developers understand an application's behavior across the different components and services within the application, as well as across other applications. The accompanying image shows an example of a serverless transaction based on asynchronous communication between several services. The right side of the image shows an in-depth look at one of the functions and the change in local variables.

How do I troubleshoot and debug my applications as productively and cost effectively as possible?

Key Features

	Real-time Debugging	Post-execution Debugging	Observability
Post execution analytics without pausing the execution		✓	✓
During execution analytics by pausing the execution	✓		
Visibility on single-app with local variables	✓	✓	
Visibility on event-driven architecture with end-to-end tracing			✓
Pre production method	✓	✓	✓
Post production method		✓	✓

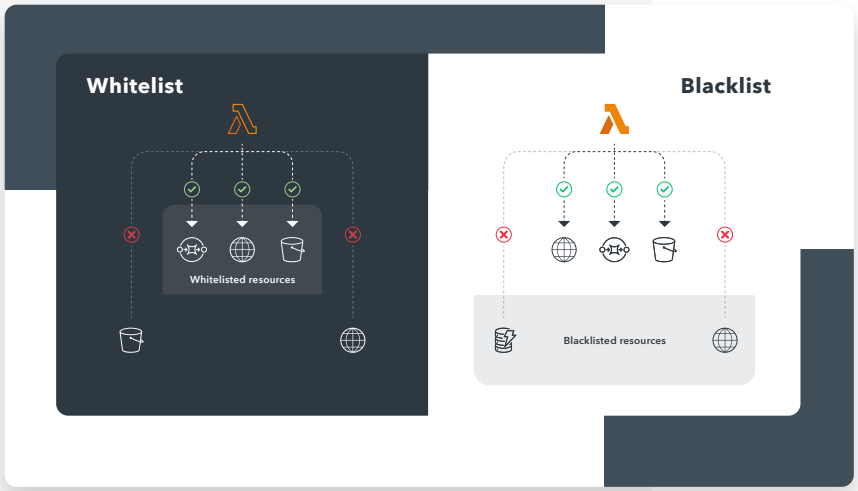
Comparison of Debugging Approaches by Thundra

Software debugging can be a challenging task for development teams adopting a serverless microservices architecture for their distributed applications. Developers who are unable to debug their actual remote live environment try to do so by simulating their remote cloud environment locally. Unfortunately, they are unable to achieve the same behaviors locally, such as simulating AWS Identity and Access Management (IAM) policies or the full set of actual services used by their applications.

To rapidly troubleshoot and debug microservices applications, teams need to bring together:

- **Real-time debugging:** This allows developers to get an in-depth look at the mechanics of a single application, understand how it's working, and how it's impacted during the execution. This is done by pausing the invocation and playing with the code and the values of local variables. It's like a native debugging experience when the application is in the developer's local environment.
- **Post-execution debugging:** This allows developers to see what happened in the code line by line after an application is executed, without pausing the execution.
- **Observability of event-driven applications** with logs, metrics, and distributed tracing together: This helps developers understand the end-to-end behavior of their applications and keep track of service-level metrics such as throughput, health, and latency.

Download the whitepaper from Thundra to dig deeper into serverless debugging.



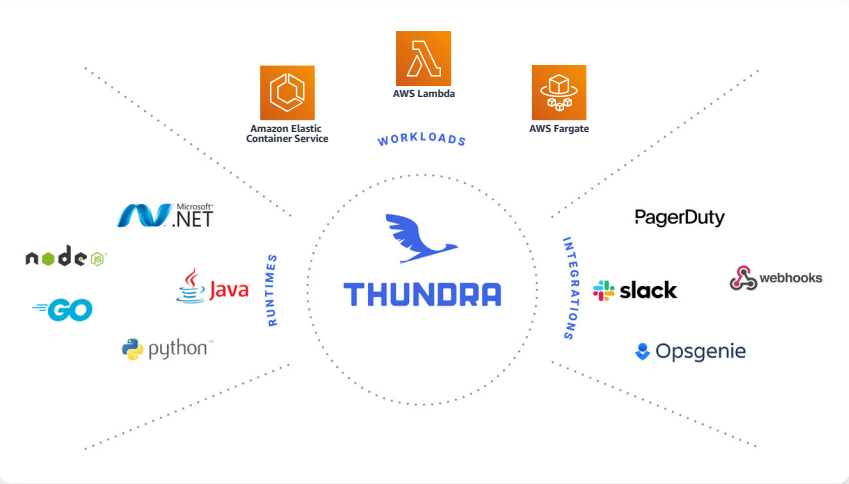
Security Policies for Controlling Outbound Traffic

LEARN MORE **about Thundra's security capabilities.**

How do I ensure my applications meet company requirements from a security, compliance, and architectural best practices perspective?

Security teams traditionally put controls in place to limit the resources to which everyone has access. They're challenged with ensuring that developers are consuming the right services and that "bad actors" aren't making calls and utilizing services that may be impacting business from a cost, security, and compliance perspective. This is challenging with cloud-native microservices applications.

Increasingly, security teams are seeking to shift risk to the left, from production to development and testing, and treat security and compliance policy as code. They are leaning on IAM permissions for serverless functions and security groups for any application. In addition, they are using a traditional whitelisting or blacklisting configuration for applications to ensure which resources it can communicate with and to help control outbound traffic.



Thundra Everywhere

Achieve application health by better managing application behavior using Thundra on AWS

You can better understand complexities and manage large-scale microservices applications across development, testing, staging, and production environments using Thundra on Amazon Web Services (AWS).

The Thundra Application Observability and Security Platform combines distributed and local tracing with debugging for serverless workloads.

Easily troubleshoot your applications with a high-level distributed view from the local service level all the way down to the line of code that was executed in your runtime environment and see the specific variables that changed.

LEARN MORE **about Thundra.**

May 2020