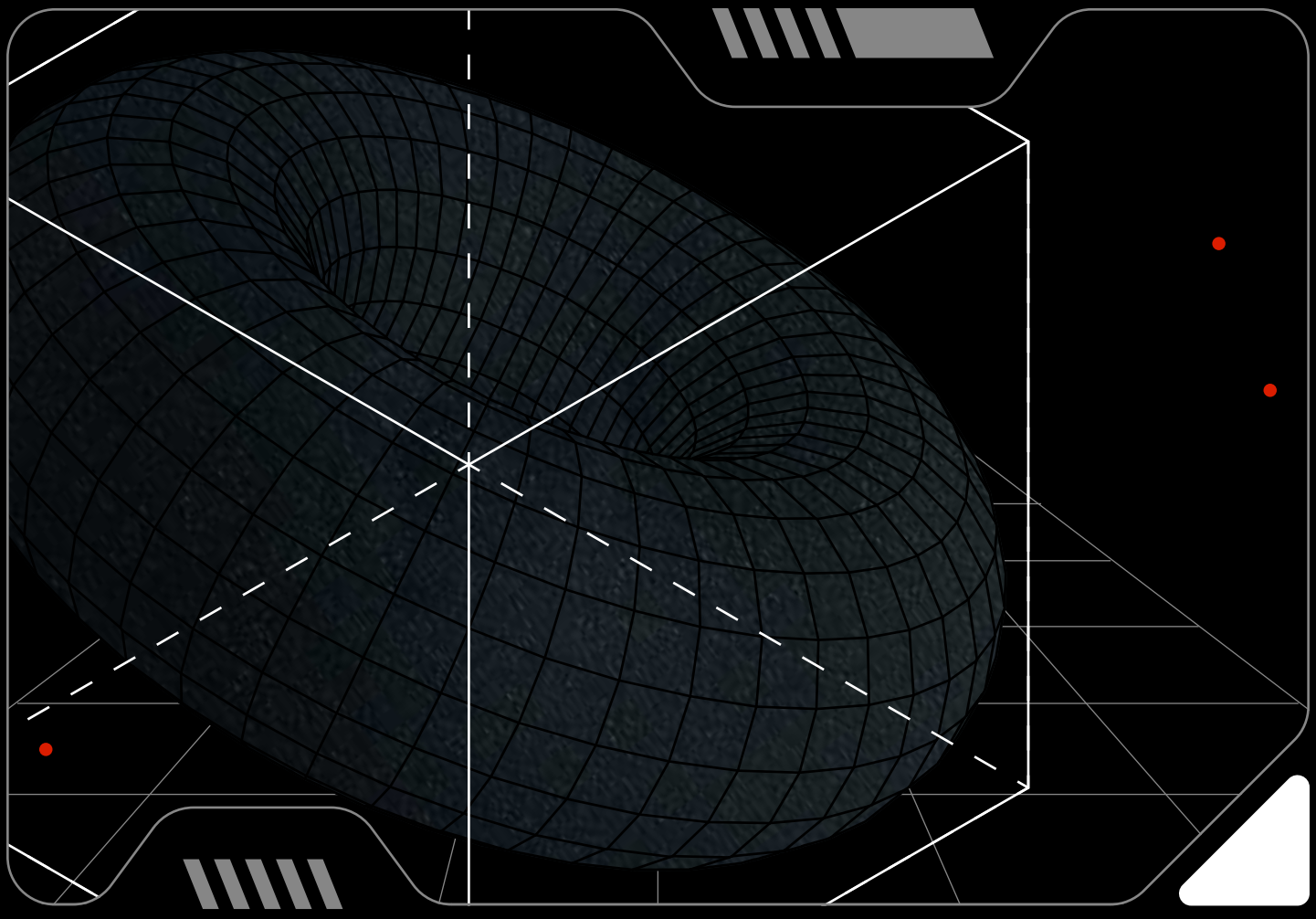
OLIGO

[E-Book 2026]

Runtime Workload Protection *for Cloud & Kubernetes*

Why static security controls leave production workloads exposed—and how runtime visibility changes the equation.



oligo.security

[1]

The Reality Check

Security teams in modern organizations live in a paradox. They deploy scanners and cloud security platforms to find misconfigurations, vulnerabilities, and active threats, yet the number of "critical" findings keeps growing faster than their engineering capacity.

The data reflects the frustration. Practitioner research shows that 53% of security professionals now rank Application Detection and Response among their most requested capabilities, and 65% want better visibility into how applications behave at runtime. Many teams are dissatisfied because posture-heavy tools generate list after list of potential risks without indicating which ones actually matter.

53%
Rank ADR among most requested capabilities

65%
Want better visibility into application behavior at runtime

The challenge is especially acute in cloud and Kubernetes environments. Containers and functions spin up and down in seconds, and runtime states frequently diverge from scanned images. Configuration drift, malicious code inside images, application exploits, kernel attacks, and exposed secrets materialize only while workloads are running. Static scanners cannot keep pace with these dynamic micro-services and therefore produce long lists of theoretical risks.

Because Oligo focuses on the application in production, it can observe exactly what happens when applications run and whether vulnerable code is actually loaded and executed. While defenders chase theoretical risks, attackers exploit running code. The assumption that pre-deployment scanning alone can secure production workloads is flawed: it amplifies noise and slows teams. Real runtime insight is the missing piece.

[2]

Why Existing Approaches Fall Short

Security teams are not failing for lack of tools. They are failing because their tools answer the wrong question. The question is not "What vulnerabilities exist?" The question is "Which vulnerabilities are actually exploitable right now?"

Static scanning lacks runtime context

Dependency scanners and vulnerability management tools identify thousands of CVEs but cannot tell whether the vulnerable library is loaded or whether any of its functions are executed in production. At Sage, the security team collected vast numbers of findings from Dependabot but had no way to determine which vulnerable libraries were even loaded in their running environment. They needed runtime context to know which vulnerabilities could actually affect them. Without that context, teams chase theoretical risks indefinitely.

AI-powered attacks blend into the noise

Attackers now wield AI to mutate malware that hides in application user space, evading detection. The recent "VoidLink" campaign illustrates the tactic: it slips into layers where EDR and CSPM operate, encrypts itself, and vanishes before signature-based tools can react. Such dynamic threats exploit the lag in traditional point-in-time scanning, underscoring the need for real-time runtime visibility beyond what conventional approaches offer.

Posture and CNAPP tools emphasize potential risk, not active exposure

CNAPP platforms unify cloud security posture management (CSPM), workload scanning, and identity governance, but they still rely on static signals. They treat misconfigurations, vulnerabilities, and identity issues with equal weight, overwhelming teams with alerts while failing to distinguish between issues that are exercised in production and those that are purely theoretical. Organizations report being satisfied with posture visibility yet still lacking runtime and application-layer protection.

CNAPP sensors cannot see inside the application

Many runtime modules within CNAPPs capture system calls at the host level but lack the ability to inspect the application's call graph or determine whether a vulnerable function is actually executed. As a result, they cannot distinguish between a loaded library and executed code, leaving security teams blind to the 1% of vulnerabilities that are truly exploitable. Oligo monitors the application itself to track which functions are called, delivering evidence-based prioritization and dramatically reducing noise.

Infrastructure-centric tools miss the application layer

Traditional cloud workload and runtime security tools take an infrastructure-centric view, looking mainly at hosts and system events. Yet many attacks originate at the application layer, exploiting weaknesses in running code that are invisible to tools focused on the operating system or container level. Without insight into which libraries or functions are actually executed, these solutions produce generic alerts and lack the context needed to separate real threats from dormant vulnerabilities.

[3]

What Actually Needs to Change

Modern cloud security requires a shift from static detection to runtime assurance. Teams must move from dealing with potential risks to asking "What is happening right now?" and "Which vulnerabilities are actually loaded and executed?" That means monitoring workloads to observe application activity in production and correlating that context with posture, vulnerability, and identity data.

Analysts have labeled this evolution Cloud Application Detection and Response (CADR): a model that unifies runtime visibility, application context, and response capabilities. By examining function-level reachability and actual execution, security teams can prioritize the vulnerabilities that matter and ignore the rest. This evidence-based approach improves trust between security and engineering and enables faster remediation.

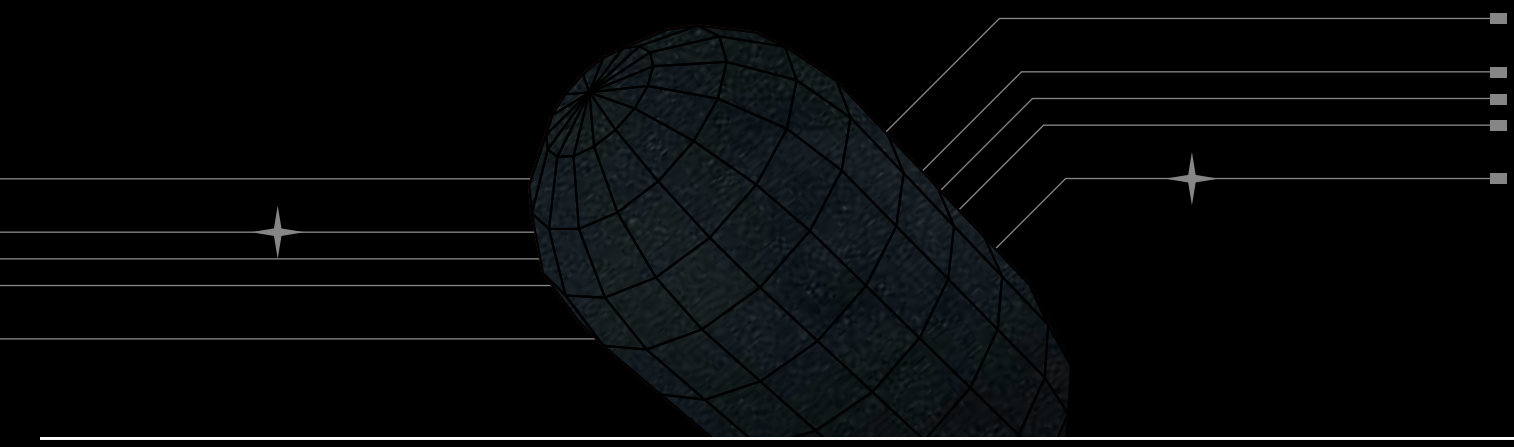
[4]

The Runtime Security Model

Runtime security correlates network, container, OS, and application-layer telemetry into a coherent view. Rather than scanning code periodically, it continuously monitors running workloads to observe which libraries are loaded, which functions are executed, and how they interact with the host kernel.

Deep code and OS-level observation enables detection of anomalies including configuration drift, malicious code, application exploits, kernel escapes, and secret exposures. With rich context, runtime security separates real threats from noise and enables active defense, blocking attacks in real time without disrupting critical services.

The result is unified visibility across cloud, workloads, applications, and AI processes. Security teams understand not just when attacks happen, but root causes and how to stop them before post-exploitation activity begins. This shared visibility aligns AppSec, Cloud Security, and DevOps around what actually happens in production, rather than what was hypothesized during a scan.



[5]

How Oligo Delivers This

Purpose-built for cloud and Kubernetes workloads

Oligo is designed for cloud-native environments where containerized micro-services and serverless functions are spun up and down continuously. It deploys in minutes for modern cloud applications built on Kubernetes or older applications hosted on-premises, inspecting the application's runtime regardless of the underlying platform. This makes it uniquely capable of defending production workloads across hybrid environments without requiring code changes or intrusive instrumentation.

Deep visibility into the full attack lifecycle

Oligo provides in-depth visibility into actual execution across the full incident lifecycle: processes, network, file and syscall activity, and function-level call stacks. This depth allows Oligo to detect fileless malware execution, attempted container escapes, kernel exploits and privilege escalations, unauthorized access to Kubernetes secrets, misuse of service account tokens to communicate with the API server, and other sophisticated attacks.

By pairing host telemetry with deep application inspection, Oligo maps every syscall to the specific functions and code paths executed in your application, enabling precise, context-rich detections while suppressing noise from unused libraries and dormant vulnerabilities. The result is clearer signal that shortens mean time to resolution.

Non-intrusive deployment and minimal overhead

Oligo's runtime sensor deploys via a simple YAML file. Sage's engineers installed it within an hour and began seeing meaningful metrics immediately. Because the sensor is lightweight and requires no code changes, developers reported that they "don't want to turn it off." Oligo runs smoothly across Kubernetes clusters and cloud services without impacting performance.

Runtime evidence versus scanning noise

While traditional cloud security tools unify scanning and posture management, they still rely on static signals and produce a high volume of theoretical findings. Oligo acts as the final authority on risk, validating whether a flagged vulnerability is loaded and executed. Teams can use CNAPPs for broad environment visibility and then rely on Oligo as a runtime validation layer, ensuring that engineering only receives issues that are actually exploitable. One customer achieved a 99% reduction in noise across hundreds of thousands of findings.

[6]

Enterprise Outcomes

Reduced workload risk and faster triage

At Sage, Oligo reduced actionable vulnerability findings by nearly 90% by proving that only 10% of identified vulnerabilities were loaded and executed in production. Engineers were saved from unnecessary patching cycles and could focus on higher-priority work. A technology company with hundreds of thousands of vulnerabilities found that only a few hundred had executed dependencies, meaning roughly 99% of their findings were noise.

90%
Reduced actionable vulnerability findings

10%
Vulnerabilities were loaded and executed in production

99%
Findings were noise

Faster investigations with root cause analysis

With application telemetry, Oligo delivers the context cloud security teams need to understand whether workload anomalies are tied to application activity. Investigations that previously required hours of manual correlation can be completed in minutes with evidence already mapped to specific functions and code paths.

Improved developer productivity and trust

Developers appreciate that Oligo eliminates false positives. In Sage's case, engineers kept the sensor deployed because it worked smoothly and saved them time. Another organization reported that Oligo strengthened the relationship between security and engineering by ensuring only runtime-validated issues reached the development queue.

Operational stability

Oligo's lightweight agent runs reliably without causing production slowdowns or outages. Reliability and security can coexist, giving teams confidence that runtime protection will not disrupt business operations. This is particularly critical in regulated industries where uptime and rapid remediation are non-negotiable.

[7]

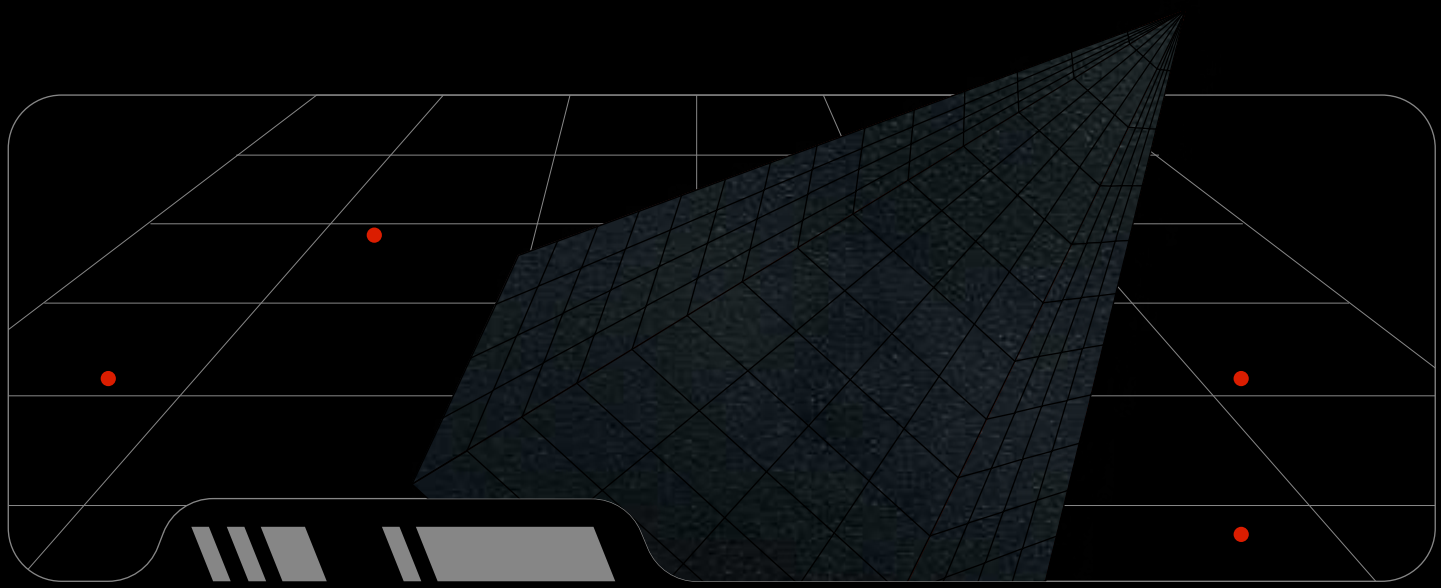
Where Oligo Fits in the Stack

Oligo works in tandem with your existing cloud security and posture-management solutions rather than displacing them. Posture-management tools effectively identify misconfigurations, known vulnerabilities, and environmental exposures across cloud and Kubernetes environments. What they cannot do is tell you what is actually exploitable at runtime.

By integrating Oligo, teams can validate whether a flagged library is truly active and whether its vulnerable functions are executed. In practice, this partnership helps prioritize genuine threats and allows security and DevOps teams to focus remediation on issues that pose real risk, enhancing the entire cloud-native protection stack without replacing critical tooling. Oligo becomes the runtime validation gate, ensuring that only credible, exploitable issues get escalated.

For teams that already invest in CNAPP platforms, the value proposition is straightforward: those tools give you broad visibility; Oligo tells you what is true. Traditional security tools expose components across your environment but cannot determine whether those vulnerabilities are ever loaded or exercised. Oligo provides the missing runtime proof, checking whether CNAPP-flagged libraries are loaded and whether their vulnerable functions are executed before any engineering work begins. The combination reduces noise and strengthens trust between security and engineering.

Because Oligo monitors the application process directly, it can also detect and block exploitation in real time. Posture tools cannot do this. This capability aligns with modern threat patterns: attackers increasingly target application-layer vulnerabilities within containers and microservices, move laterally through dynamic workloads, and hide malicious code inside images. Oligo's runtime sensors monitor such events in real time and block malicious or suspicious activity. It is your last line of defense and your source of truth when facing sophisticated cloud-native attacks.



[8]

When This Matters Most

Runtime protection becomes essential in four distinct scenarios:

1

When your organization runs Kubernetes or containerized workloads in production, where dynamic micro-services change faster than static security controls can keep up.

2

When vulnerability backlogs and alert fatigue overwhelm your Cloud Security program; Oligo cuts through noise and focuses attention on exploitable issues.

3

When you rely on CNAPP platforms for posture management but still lack runtime context; Oligo provides the missing validation layer.

4

When you operate in regulated industries where uptime and rapid remediation are critical; Oligo's non-intrusive deployment and reliable performance support both compliance and stability.

[9]

Proof in Production

The best measure of a security tool is what it does for the teams that deploy it.

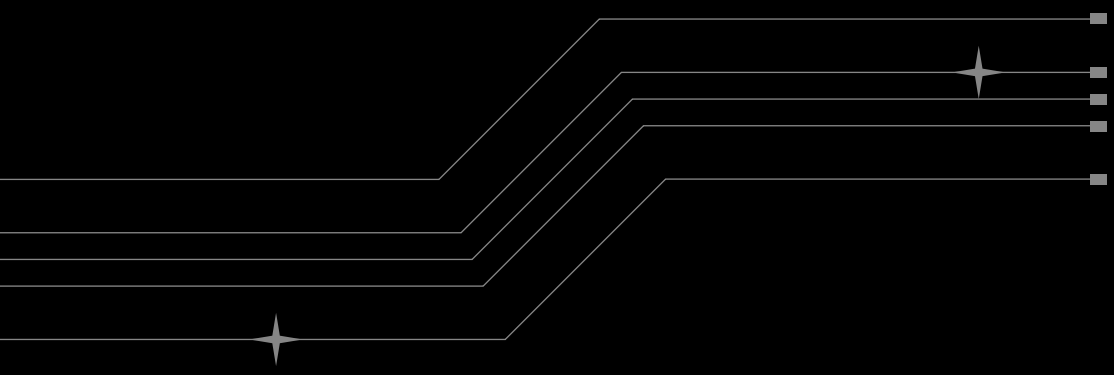
> **"To address our findings effectively, we needed runtime context and insights to help us identify the vulnerabilities that can actually affect us." — Javan Rasokat, Sr. Security Specialist, Sage**

After deploying Oligo, Sage's team reduced their vulnerability backlog by nearly 90% and saved developers so much time that engineers didn't want to turn it off.

> **"Oligo gives us a reliable source of truth. It's the final authority we use before something becomes engineering work."**

Using their CNAPP for broad visibility and Oligo for runtime validation, this customer narrowed hundreds of thousands of findings down to a few hundred, a 99% reduction in actionable noise.

> **"Runtime is where reality lives. If we had to defend one tool, it would be the one that tells us what's actually true."**



OLIGO

Oligo sees running code to prioritize, detect, and prevent attacks.

oligo.security