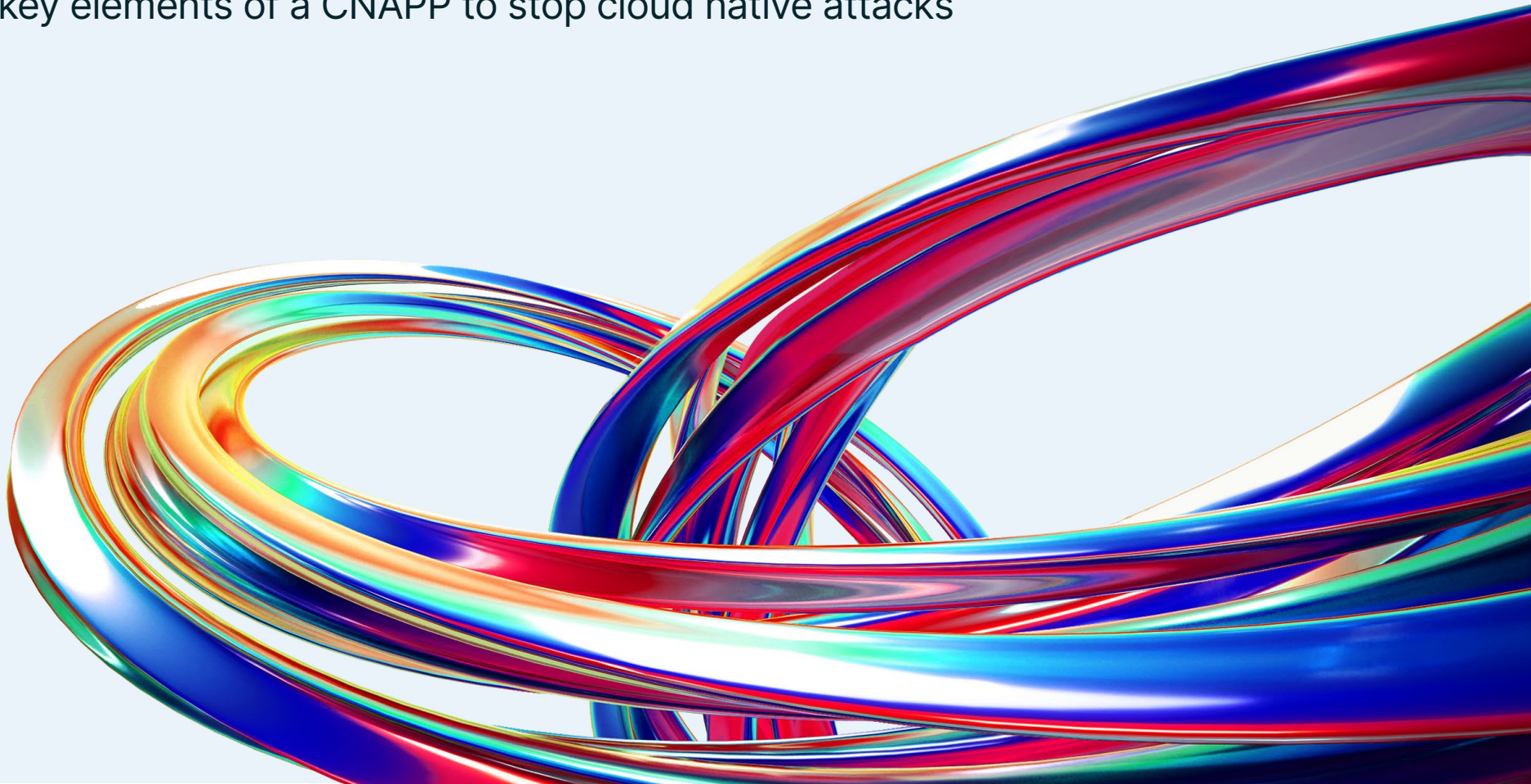




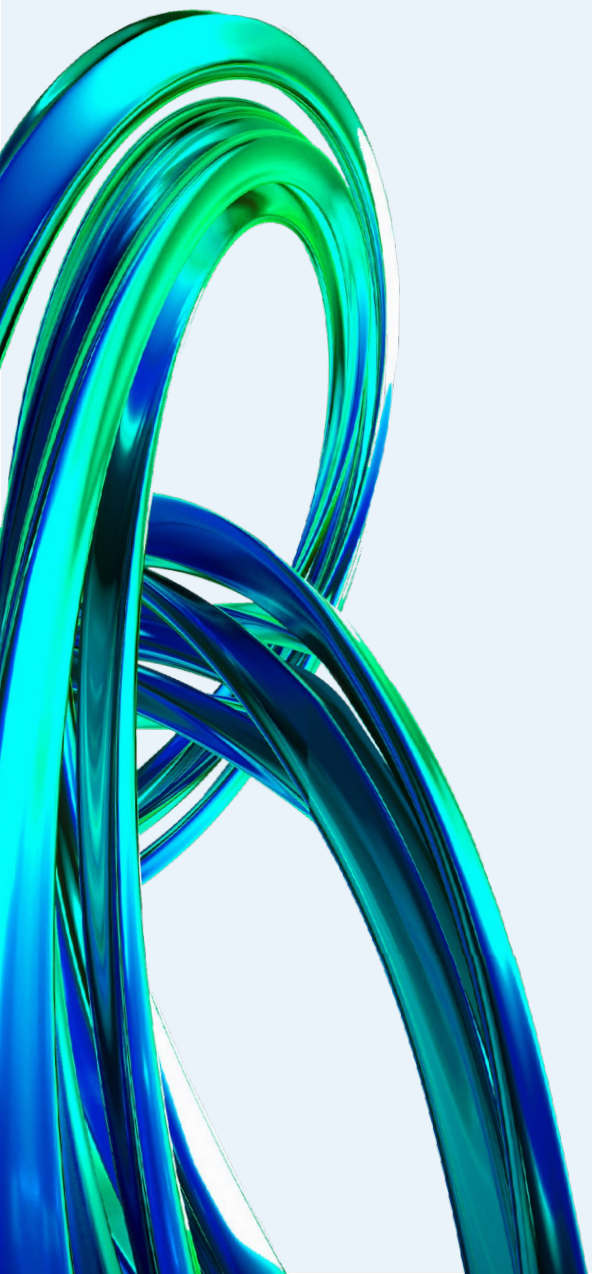
CNAPP 101

Achieving Cloud Native Security

The key elements of a CNAPP to stop cloud native attacks



Contents

- 
- 03** Introduction
 - 04** Element 1: Open source and the software supply chain
 - 05** Element 2: CI/CD pipeline
 - 06** Element 3: Microservices
 - 07** Element 4: Immutability
 - 08** Element 5: Orchestration
 - 10** Element 6: 'Run-anywhere' stack
 - 11** Element 7: Application context
 - 12** Why traditional security approaches won't work for cloud native applications
 - 13** CNAPP checklist
 - 14** Prevent attacks before they happen, stop them as they happen

Introduction

What is Cloud Native?

According to the [CNCF](#), Cloud native technologies empower organizations to build and run scalable applications in modern, dynamic environments such as public, private, and hybrid clouds. Containers, service meshes, microservices, immutable infrastructure, and declarative APIs exemplify this approach.

These techniques enable loosely coupled systems that are resilient, manageable, and observable. Combined with robust automation, they allow engineers to make high-impact changes frequently and predictably with minimal toil.

A cloud native application is designed to run on a cloud native infrastructure platform and is resilient, agile, operable, and observable.

What is CNAPP?

In Gartner's recent [CNAPP Market Guide](#), CNAPP is defined as, "a unified and tightly integrated set of security and compliance capabilities designed to secure and protect cloud-native applications across development and production.

CNAPPs consolidate a large number of previously siloed capabilities, including container scanning, cloud security posture management, infrastructure as code scanning, cloud infrastructure entitlement management, runtime cloud workload protection and runtime vulnerability/configuration scanning."

Let's break down the CNAPP acronym backwards:



Platform

A complete platform to secure the entire application life cycle



Protection

Offers prevention and response capabilities, rather than just visibility



Cloud native applications

Traditional application security isn't applicable

Element 1

Open source and the software supply chain

Open source is everywhere

Cloud native brings a huge shift to the way modern applications are built by using a large majority of open source components and a small amount of custom code. Chances are high that open source is already present in your codebase.

97% of today's codebases that were audited contain **open source components**

2022 OSSRA Report

But open source isn't the only component being assembled into modern applications. The reality of the cloud native software supply chain is that organizations rely heavily on third-party code, potentially introducing vulnerable, risky, unlicensed, or out-of-date components into their applications. Many times, even base images are obtained from third parties. Through malicious images and code, attackers can target deployments and potentially gain access to the host and the application running on it.

What to do?

Identify and manage your risk exposure across the whole software supply chain – not just open source software and vulnerabilities

Gain comprehensive visibility into and control over open source components.

Proactively identify security vulnerabilities before applications are released into production. Software composition analysis (SCA) helps you to evaluate and manage the risks caused by open source vulnerabilities in third-party dependencies. Understand the places in your software supply chain where third-party components are being used and put runtime controls into place that can stop malicious activity that wasn't detected early on.

Element 2

CI/CD pipeline

Move fast to reduce the attack surface

The CI/CD process is critical to accelerate the software development cycle, allowing organizations to quickly add features that address evolving customer needs. Instead of months, every stage now takes hours or even minutes, and new releases can be deployed several times a day.

Increased speed also creates a more dynamic environment with opportunities to introduce security risks. Speed can take precedence over security, and developers can unknowingly put applications at risk.

What to do?

'Shift left' to prevent attacks before they happen
Automate testing in the developer workflow —
don't hinder speed and agility

With the dynamic delivery cycles and increased number of releases, automation is crucial to producing applications with fewer flaws at a high speed. The cloud native approach relies on strong automation from the very start, making high-velocity CI/CD pipelines possible.

Advice from the experts

Enable DevSecOps

As organizations are shifting left, they need to make security part of development from the start and encourage day-to-day collaboration between developers and security teams. To bridge security and agility, the DevSecOps approach embeds security at the beginning and throughout every stage of the software development life cycle.

Just shifting left won't make it right

Shift left is designed to address security issues as early as possible, significantly reducing the attack surface. However, it doesn't prevent exploits or attacks in runtime and won't let you know if an attack was attempted. It doesn't provide visibility into the application or block suspicious activity in real time.

Element 3

Microservices

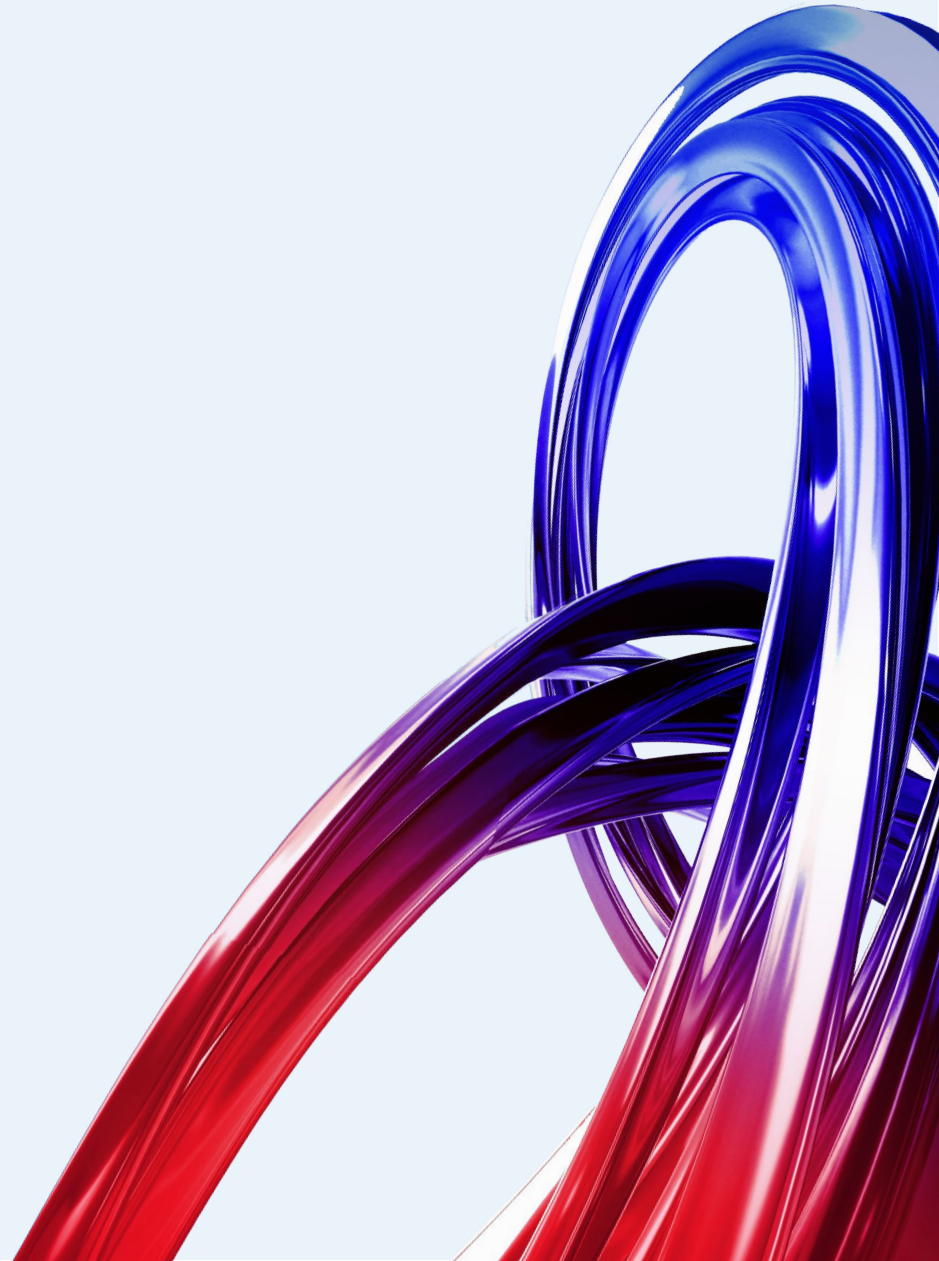
Microservices have emerged as the architecture of choice for cloud native applications. In contrast to monolithic architectures, where every piece of the application is intertwined, a microservices-based application is broken up into smaller pieces that can be developed and managed independently. The discrete pieces are integrated via APIs in a loosely coupled environment. Each microservice has well-defined boundaries of functionality, which makes them easier to understand and predict.

Securing the application at the microservice level could prevent an attack from spreading much sooner, and restrict it to a much smaller radius, than was ever possible before.

What to do?

| Focus on the expected behavior of microservices

Microservices divide product functionality into units that can be deployed independently. This approach is much more predictable since each service represents a functional block with a well-defined interface that does only one thing. With applications broken down into microservices that perform a single function, it's more effective to base security controls on their intended activity as opposed to trying to correlate complex interactions between components.



Element 4

Immutability

One of the key aspects of cloud native is the concept of immutability. In a traditional environment, with a monolithic application running on a VM, developers usually make changes by remotely logging in to the machine or pushing code changes manually. Containers, however, are designed to be immutable, meaning they don't change once they're deployed.

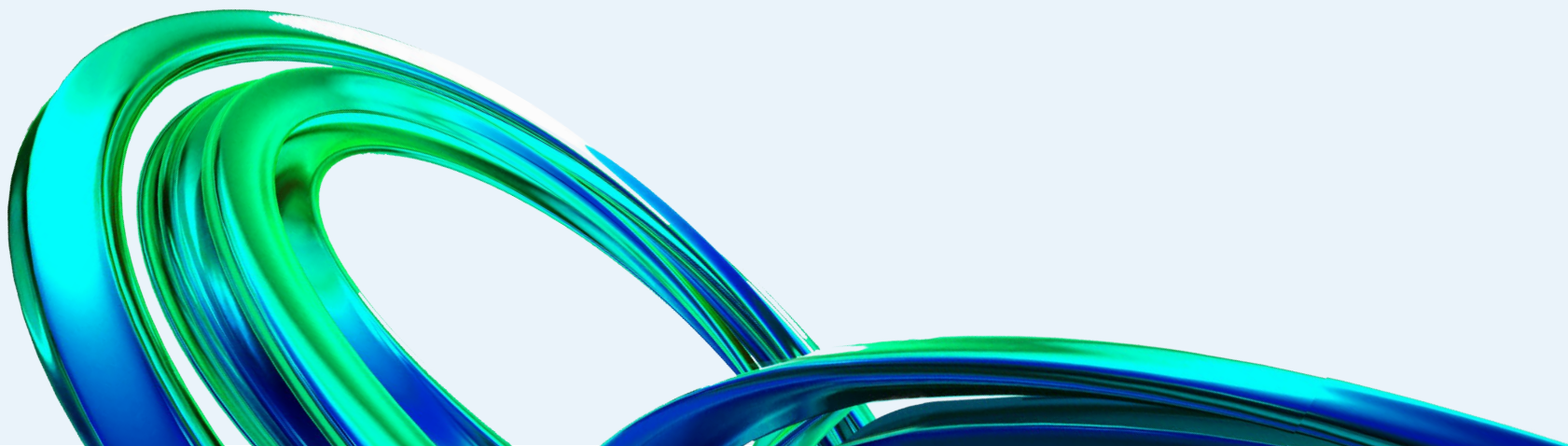
A container should not be modified in runtime: no updates, no patches, no configuration changes. When you need to update a deployed container, instead of changing it in runtime or accessing the machine via Secure Shell (SSH), you build and deploy a new image.

What to do?

Enforce immutability

When containers can't be updated live, it's easy to determine suspicious behavior, by blocking anything that wasn't part of the original image. Immutability makes security deterministic. When a container is modified, it automatically means that it's been compromised.

By tracking a workload from its inception to runtime, you can compare its current payload to its original state to identify every difference and block those differences from execution. Enforcing immutability by disallowing changes to running workloads helps to reduce the overall attack surface and provides a solid foundation for the security of your cloud native applications.



Element 5

Orchestration

Managing containers at scale

Unlike physical servers or VMs, containers are by nature lightweight and ephemeral. They can be dynamically spun up and down, stopped and destroyed at any moment, then rebuilt and replaced with minimal setup. They are short-lived, often lasting only for minutes.

Combined with frequent release cycles, running such ephemeral workloads in production can quickly become a challenge, especially at scale. With each microservice typically hosted in its own container, enterprise cloud native deployments can span hundreds, thousands, or even tens of thousands of containers running in production. To manage the complexity of operating containerized workloads, automation via orchestration is essential. That is why organizations rely on orchestration tools, such as Kubernetes, which fully automate the deployment, scaling, management, and networking of container-based applications.



With more and more organizations adopting Kubernetes for mission-critical workloads, we are seeing an increase in concern around the security of cloud native applications.

When delivering cloud native applications, organizations must balance these concerns with the need for speed and agility

[Help Net Security](#)

Element 5

Orchestration

What to do?

Protect workloads in real time

To protect ephemeral cloud native workloads, security controls need to be as agile as the containers or functions themselves and be deployed close to the application to enforce policies accurately. Security must follow workloads wherever they run and move fluidly from anywhere to anywhere. They must also be governed in a consistent, holistic manner and be visible and controlled through a single platform. This approach enables granularity and continuity while ensuring that workloads are protected at any time.

Identity-based segmentation

By implementing identity-based segmentation with granular controls, you make it harder for bad actors to propagate throughout your network, and you reduce the potential attack surface.

Advice from the experts

Hardening isn't enough

In a multi-cloud environment, at the scale of organizations' cloud native footprint today, the sheer number of required configurations is daunting for any administrator. Human error must be accounted for, and automated configuration checks are critical for both orchestrators and cloud accounts. This approach doesn't consider such things as application layer vulnerabilities and zero-day attacks.

Using only runtime security won't do

Security offerings that focus only on runtime protection are missing the critical shift in how applications and the workloads that host them are being developed – with a lot of open source components and, hence, potential vulnerabilities.

Beware of segmentation alone

Segmentation is an important step for containing an attack and restricting the potential damage. However, a segmentation-only approach doesn't offer protection from a wide range of threats, such as application-based attacks, supply chain attacks, and data theft.

Element 6

'Run-anywhere' stack

One of the biggest benefits of containers is that they're built to run in any environment. They include everything they need to run: code, system tools, libraries, and frameworks. And they can be deployed reliably across any environment, including local machines, on-premises servers, and public clouds.

The portable, "run anywhere" nature of containers can speed up delivery, while also supporting the adoption of a multi-cloud strategy. The ecosystem of cloud native tools is constantly expanding, giving organizations a seemingly endless choice of orchestration platforms, CI/CD tools, and cloud services. As a result, environments have become more diverse than ever before.

To protect your applications consistently no matter where they run, you must secure the workloads independently from the infrastructure while at the same time hardening your host, orchestrator, and cloud platform.

What to do?

■ Automation

With the high speed of DevOps, it's impossible to implement efficient security controls without automation both before and during production. The scale and dynamic nature are too much to try to keep up by using static or manually updated tools. Your security tool should keep pace with rapid delivery and production cycles by enabling automated and continual security testing as part of the CI/CD workflow. When security is built into the DevOps process, it's possible to uncover and fix security issues at the inception, thus avoiding any roadblocks for moving applications forward.



Element 7

Application context

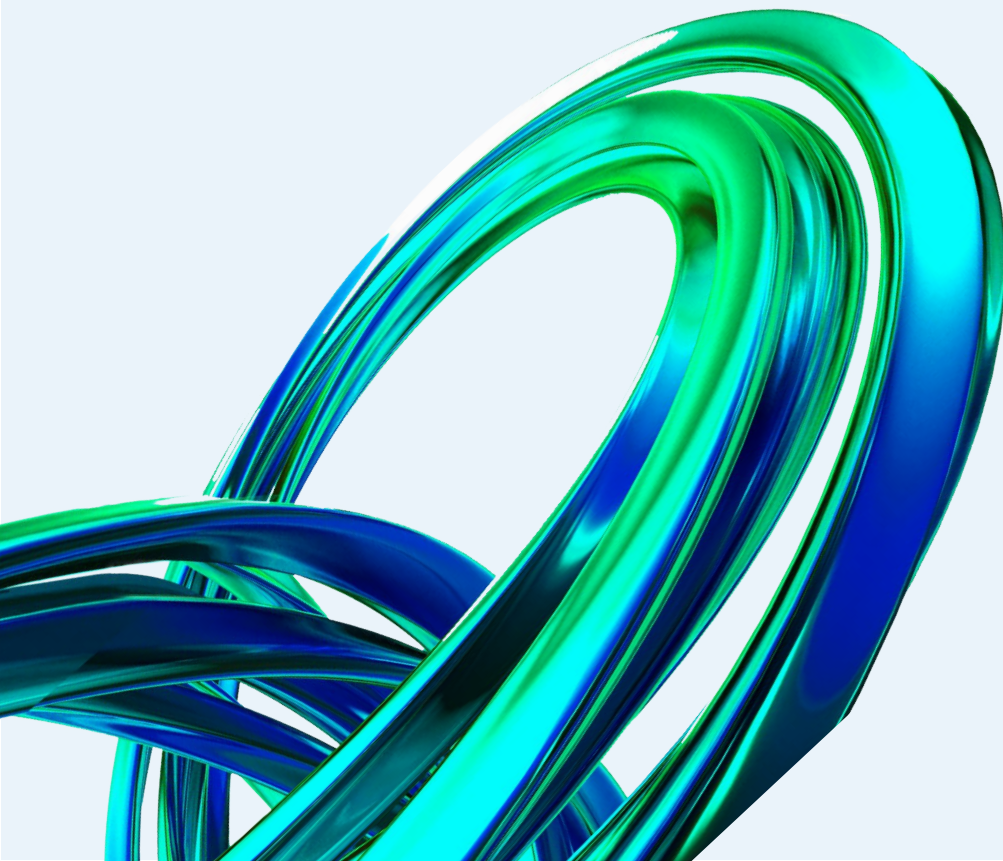
Immutable workloads and the consistency between the development and production environments allow security to maintain application context. This means that information acquired during the build is applied in the runtime environment as well. For example, scanning and signing artifacts in the CI/CD pipeline will contribute to granular decisions about their deployment – for example, preventing unvetted images from running in the production environment while detecting and blocking anomalies in runtime.

What to do?

I Apply security in context of the application

Application context implies applying policies and controls that consider risk-related contextual factors for your environments. For example, you can prioritize mitigation efforts for vulnerabilities associated with your business-critical applications by filtering them by repository name, registry, cluster, or namespace.

Leveraging data acquired across different stages of the application life cycle and the production environment allows you to maintain rich application context, which enhances security and enables teams to focus on what matters.



Why traditional security approaches won't work for cloud native applications

Host-based

Security solutions that use installed host-based agents lack the application context and appropriate control points in the cloud native stack. Without these capabilities, it's impossible to adequately detect threats and respond to them.

Network-based

Traditional network-based security tools weren't built for cloud native traffic and have limited capabilities in dynamically orchestrated environments. Cloud native networking is more complex and dynamic, with rapid reconfiguration of resources, rendering traditional network security tools insufficient and ineffective.

Endpoint detection and response (EDR)

In a distributed microservices architecture, it's virtually impossible to perform incident response effectively due to the ephemeral nature of workloads, so attacks might go unnoticed.

Cloud provider tools

All major cloud providers offer security hubs that offer a comprehensive view of customer services. While these offerings can help organizations strengthen their security posture, they're limited in scope and don't provide a single-pane-of-glass view across all your environments.

Do it yourself (DIY)

Organizations of all sizes are adopting cloud native technologies and moving their workloads to the cloud. However, there's still a significant lack of qualified, skilled professionals to support this transition, especially in non-tech-related industries. The current pace of cloud innovation and the need to evolve make it a challenge to keep the workforce relevant over time.

Given the severe shortage of cloud-native-savvy staff, building a solution to secure this new attack surface completely in-house isn't a viable option for most organizations. Aside from pioneers such as Netflix and Google that developed internal tools and services for their extremely large and mature environments, most teams would quickly be overwhelmed by the challenges of designing and maintaining a DIY solution, defeating the purpose of cloud native to develop faster.

CNAPP checklist

Holistic cloud native security

Cloud native security solutions need to address the whole application life cycle, not just isolated points along the way.

Prevent attacks before they happen

Combine scanning and automated prevention at key acceptance gates in the code, build and deployment phases to reduce the largest attack surface before production.

Automation

To keep up with the scale and speed of delivery, enable automated and continual security testing as part of the CI/CD workflow as well as runtime to fix security issues at the inception and block drift or malicious behavior in runtime.

Image assurance

Control what you build and deploy using a combination of assurance policies with acceptance gates that only allow compliant images.

Drift prevention

Ensure immutability of container workloads by enabling drift prevention at runtime. Drift prevention ensures that your workloads are protected from vulnerabilities, zero-day exploits, and internal threats that can't be caught early on in the life cycle.

Optimization for context

Maintain the context of the entire application between different stages of the application life cycle and production environment.

Granular visibility and response

Monitor running workloads and ensure the continuity of applications by blocking any suspicious container activity, without container downtime or restarts.

'Secure once, run anywhere'

Secure applications across the cloud stack no matter where and how they're deployed. Apply automated, consistent security controls to images, containers, nodes, and clusters across any orchestration platform, on Linux and Windows, and across any cloud provider.

Truly cloud native

Orchestrate across on-site deployments, public clouds, and hybrid deployments without the need to reconfigure your controls and policies for each environment.

Cloud scale

Securely scale in and out with no limitations, supporting multiple users and teams across hundreds of deployments spanning thousands of nodes.

See what others don't. Stop what others can't

The Aqua Platform is the most integrated CNAPP to stop attacks on your cloud native applications. With Aqua, DevOps and security teams can prioritize risks in minutes across the entire development life cycle and automate prevention to secure their cloud native applications on day one. Real cloud native attacks are stopped immediately without killing workloads.

With a platform built on the best-loved open source cloud native community and innovation from dedicated threat research, Aqua is a complete solution to cloud native security for transformational teams.

Secure from day one

Prioritize risks in minutes

Get a view of the top risks for your cloud native applications in minutes with a searchable asset inventory and context-based insights from across the application development life cycle.

Automate prevention

Reduce the attack surface with automated pre-production acceptance gates that prevent malicious source code, non-compliant images, infrastructure-as-code templates, and misconfigured Kubernetes workloads from getting into production.

Protect in real time

Respond instantly

use behavioral indicators based on real-world cloud native attacks surgically block attacks in production without killing workloads that others can't see. Runtime policies provide real-time protection for containers, VMs, and serverless workloads while malicious activity in the build is detected and stopped.

Integrate with your existing tools

Integrate your existing environment and stack with nearly every CI/CD, security information and event management, monitoring, and collaboration tool. Because Aqua is a certified technology partner of all major cloud providers, it also tightly integrates into your cloud setup and toolchain.

About Aqua Security

Aqua Security stops cloud native attacks across the application lifecycle and is the only company with a \$1M Cloud Native Protection Warranty to guarantee it. As the pioneer in cloud native security, Aqua helps customers reduce risk while building the future of their businesses. The Aqua Platform is the industry's most integrated Cloud Native Application Protection Platform (CNAPP), protecting the application lifecycle from dev to cloud and back. Founded in 2015, Aqua is headquartered in Boston, MA and Ramat Gan, IL with Fortune 1000 customers in over 40 countries.

For more information, visit: www.aquasec.com

