CleanStart™

Beyond CNAPP

Building a Complete Software Supply Chain Security Strategy

Visibility. Verification. Governance.

Built Clean. Run Secure.

Executive Summary

The container security market has matured rapidly, with organizations adopting an expanding range of tools to secure cloud-native environments. Cloud-Native Application Protection Platforms (CNAPPs) and Cloud Security Posture Management (CSPM) platforms have become the default choice for security teams seeking greater visibility into cloud risk. They provide dashboards, generate alerts, and help security teams understand and prioritize security findings across modern cloud environments.

But they do not remediate the underlying software.

There is an often-overlooked gap between detecting software supply chain risk and eliminating it. CNAPP and CSPM platforms are designed to provide visibility and posture management. They are not remediation platforms. They cannot rebuild a vulnerable base image, replace an untrusted third-party component, or produce a newly verified software artifact after vulnerabilities have been addressed.

Beyond detection, organizations face additional challenges that traditional cloud security platforms address only partially. These include understanding exactly what software is running and where it originated, managing image sprawl across registries, and establishing a clear remediation path when vulnerabilities are identified. Together, these challenges make it difficult to reduce inherited software risk before it reaches production.

This whitepaper explains where CNAPP and CSPM provide value, where their responsibilities naturally end, and how CleanStart extends that security lifecycle. It introduces CleanSight, CleanStart's software supply chain visibility solution, and explains how CleanStart offerings work together to help organizations discover, verify, govern, and remediate software supply chain risk from software foundations through to trusted software artifacts ready for production.

The Problems That Current Platforms Do Not Solve

Before examining what CNAPP and CSPM do, it's important to understand the problems they were never designed to address. These are not edge cases or product gaps that vendors are quietly working to close. They are architectural limitations that arise because these platforms were built to secure cloud environments rather than software supply chains. These limitations are not shortcomings of the technology itself. They reflect the scope for which CNAPP and CSPM platforms were originally designed.

Problem 1

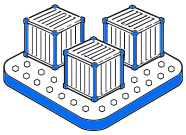
No Supply Chain-Level Visibility

Most security platforms can tell you that a vulnerability exists in a running container. Far fewer can tell you where that vulnerability originated within the software supply chain: which upstream source package introduced it, which build pipeline included it, which base image layer it resides in, and how it ultimately propagated into production.

This distinction matters because provenance determines where remediation begins. Detection without provenance is a dead end. If you cannot trace a vulnerability back to its origin in the software supply chain, you can only play whack-a-mole, patching individual instances while the underlying issue remains embedded in the build process.

Supply chain-level visibility means knowing not just what is running, but where everything came from: which packages, from which sources, assembled through which build processes, and verified with which signing and attestation artifacts. It means understanding the complete dependency graph of every container in production, including transitive dependencies several layers deep.

Most CNAPP platforms analyze the finished artifact. They are not designed to reconstruct the software supply chain that produced it.



Problem 2

Image Sprawl Is Getting Worse, Not Better

According to Sysdig's 2025 Cloud-Native Security and Usage Report, the average container image size has quintupled in recent years, driven by a 300% increase in the number of packages per image. The same report found that 87% of container images running in production environments contain at least one high or critical vulnerability.

This is not primarily a reflection of poor engineering. It is a consequence of general-purpose base images being designed for broad compatibility rather than security minimization, combined with the fact that many organizations lack a systematic way to understand and govern the software they are actually running.

The problem extends beyond the contents of individual images. Over time, container registries accumulate images that are no longer actively managed, making it increasingly difficult to distinguish real operational risk from unnecessary noise.

Traditional vulnerability scanning identifies vulnerabilities within these images, but it cannot determine whether an image is actively used, who owns it, whether it should be rebuilt, or whether it should simply be retired. Without accurate inventory intelligence, security teams spend valuable time triaging findings across obsolete artifacts instead of focusing on the software that actually supports production workloads.

Three Common Forms of Image Sprawl

UNUSED IMAGES
No longer referenced by active workloads, yet continue to consume storage and generate vulnerability findings.

DUPLICATE IMAGES
Multiple copies of the same image create inconsistent patching and duplicate remediation effort.

LEGACY IMAGES
Outdated images with large vulnerability footprints and weak or missing software provenance.

Problem 3

No Clear Remediation Path

Perhaps the most significant operational gap in current platforms is the absence of a clear remediation path. Vulnerability scanners generate findings. CNAPP platforms prioritize those findings. But neither tells an engineering team exactly what needs to change, where it needs to change, or how to ensure the vulnerability does not reappear.

In most organizations, remediation remains a manual process.

Repeated across hundreds of images and thousands of vulnerabilities

The challenge is compounded by signal-to-noise. According to Sysdig's 2025 Cloud-Native Security and Usage Report, fewer than 6% of flagged vulnerabilities are actually in use and exploitable at runtime. Yet engineering teams are often required to review every reported finding before determining which ones warrant action. Without a systematic remediation strategy, organizations spend considerable effort triaging vulnerability reports while struggling to consistently reduce the underlying software risk.

Problem 4

Third-Party Application Dependency Risk Extends Beyond Vulnerability Scanning

Container security platforms focus primarily on securing the software packaged withincontainer images. Modern applications, however, also depend on extensive application-layer ecosystems: NPM packages for JavaScript services, PyPI packages for Python applications, Maven artifacts for Java workloads, and similar package repositories across other languages. A single microservice can introduce hundreds of direct and transitive dependencies from public registries, each becoming part of the application's software supply chain.

Many CNAPP platforms can identify known vulnerabilities within application dependencies through integrated Software Composition Analysis (SCA) capabilities. However, vulnerability detection is only one part of the problem. Determining whether a dependency should be trusted requires additional context that extends beyond CVE identification.

Questions such as whether a package was recently published, whether it resembles a well- known package in a potential typosquatting attack, whether it has an unusual maintainer history, or whether the published package differs from its upstream source are indicators of software supply chain risk rather than traditional vulnerability management. Addressing those risks requires dependency governance, provenance, and verification alongside vulnerability scanning.

What CNAPP and CSPM Actually Do

To understand the remaining gaps, it is important to be precise about what these platforms were designed to do.

Cloud Security Posture Management (CSPM)

focuses on cloud infrastructure. It identifies misconfigurations such as exposed storage buckets, overly permissive IAM roles, missing encryption settings, and publicly accessible cloud resources. Its purpose is to improve cloud security posture rather than the software running inside container images.

Cloud-Native Application Protection Platforms (CNAPPs)

extend this = visibility by bringing together capabilities such as CSPM, Cloud Workload Protection (CWPP), Cloud Infrastructure Entitlement Management (CIEM), and Kubernetes Security Posture Management (KSPM). Together, these capabilities help organizations understand runtime security posture, prioritize operational risk, and respond more effectively across cloud-native environments.

These capabilities solve important operational problems. They do not, however, rebuild vulnerable container images, govern third-party dependencies, or remediate inherited software risk. Their role is to understand the security posture of software that already exists, not to determine what software enters production.

CNAPP and CSPM tell you the house is on fire. They do not put it out.

Where the Real Risk Lives Inside the Image and Dependency Layer

One of the most common misconceptions in container security is treating vulnerability scanning and a hardened image strategy as interchangeable. They are not. They operate at different stages of the software supply chain, produce different outcomes, and solve different problems.

A vulnerability scanner analyzes the contents of an existing container image and identifies known vulnerabilities by comparing installed software against vulnerability databases such as the National Vulnerability Database (NVD) and the CISA Known Exploited Vulnerabilities (KEV) Catalog. Its output is visibility into known vulnerabilities. It does not change the software contained within the image.

As new vulnerabilities are disclosed in upstream packages, that report continues to grow unless the underlying software is updated or rebuilt. Without an effective remediation strategy, vulnerability backlogs accumulate over time. This is the defining characteristic of a detect-only approach: identifying risk without reducing it.

Industry data reinforces this challenge. As discussed in Section 1, the vast majority of production container images contain high or critical vulnerabilities, not because they were built carelessly, but because they inherit software from general-purpose base images designed for broad compatibility rather than minimal attack surface. Every unnecessary package included in a container image expands the software footprint an organization must maintain and secure throughout the software lifecycle.

Kubernetes Security Posture Management (KSPM) addresses a different class of problems. It can identify workloads running as root, excessive Linux capabilities, overly permissive Pod configurations, or insecure Kubernetes resource settings. These are important runtime and configuration risks. However, KSPM is not designed to analyze the software packaged within the container image itself. A vulnerable cryptographic library, an outdated system package, or an unnecessary runtime dependency remains a software supply chain problem that originates before the workload is ever deployed.

The Three Gaps CNAPP Leaves Open

When organizations rely exclusively on a CNAPP to secure their container environments, three important gaps remain in their software supply chain security posture

Gap 1

No Remediation Path, Only a Growing List

A CNAPP identifies which vulnerabilities exist across workloads and container images, helping security teams understand and prioritize operational risk. It does not eliminate those vulnerabilities at the source. Without a remediation strategy, every newly discovered vulnerability requires engineering teams to update the underlying software, rebuild the image, validate the application, and redeploy the workload.

This process is difficult to scale across hundreds of images and thousands of vulnerabilities. As discussed earlier, Sysdig's 2025 Cloud-Native Security and Usage Report found that fewer than 6% of reported vulnerabilities are associated with packages actively loaded into memory at runtime. While every vulnerability should be evaluated within the context of an organization's risk and compliance requirements, the finding highlights a practical challenge: engineering teams often spend significant effort reviewing vulnerability reports while making limited progress toward reducing the underlying software risk.

Gap 2

Limited Governance of Third-Party Application Dependencies

Modern CNAPP platforms increasingly integrate Software Composition Analysis (SCA) capabilities to identify vulnerabilities within application dependencies. However, identifying vulnerable packages is different from governing which packages are allowed to enter the software supply chain.

Questions such as whether newly published packages should be temporarily restricted, whether a package exhibits characteristics of typosquatting, whether it originates from a trusted source, or whether the published package is consistent with its upstream source repository require dependency governance rather than vulnerability detection.

These decisions are made before software is built and deployed. By the time a dependency appears in a running workload, it has already become part of the application's software supply chain.

Gap 3

No Remediation Commitment orService-Level Assurance

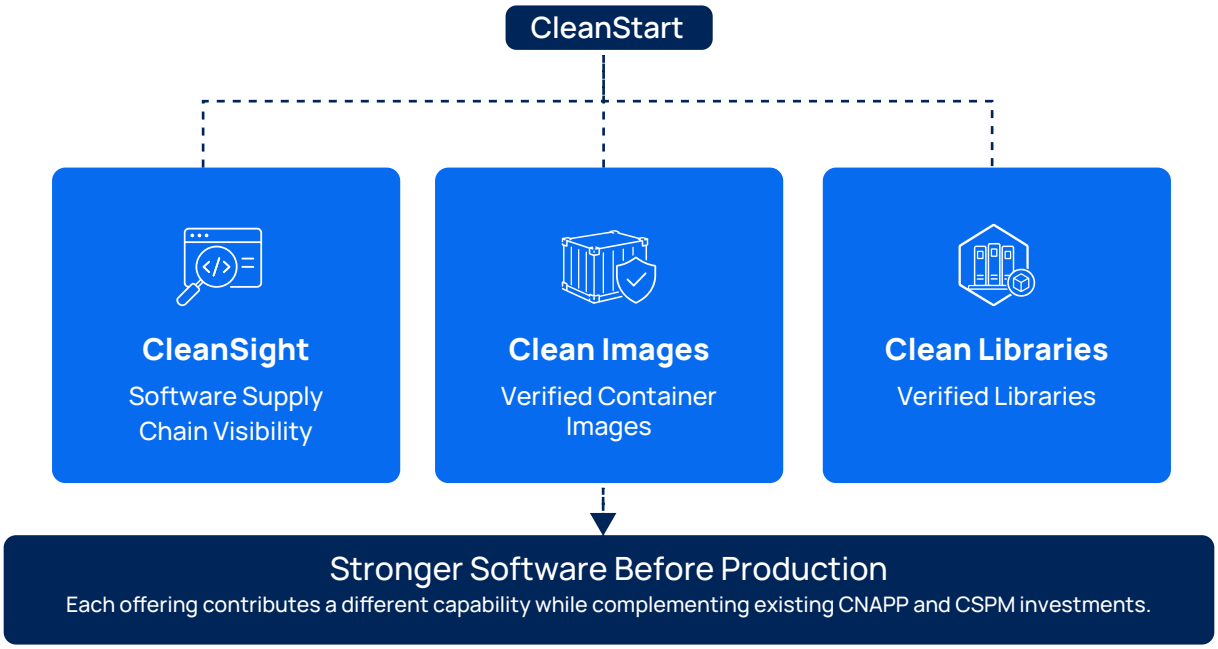
A CNAPP subscription provides visibility into security findings. It does not provide a service-level commitment for when vulnerable software will be rebuilt, updated, or replaced. When a critical vulnerability is disclosed, a CNAPP can identify the affected workloads and help prioritize response efforts. The responsibility for rebuilding images, validating updates, and deploying remediated software remains outside the scope of the CNAPP itself.

For organizations operating in regulated environments, including FedRAMP, PCI DSS v4.0, HIPAA, NIS2, and DORA, demonstrating timely remediation often requires documented operational processes, defined service-level objectives, and auditable evidence. Visibility into vulnerabilities is an important part of that process, but it is only one component of an effective remediation program.

How CleanStart Closes the Gap

Solving software supply chain risk requires more than identifying vulnerabilities. It requires understanding where software came from, verifying the software entering the build process, and reducing inherited risk before applications reach production.

CleanStart addresses these challenges through three complementary offerings that work together across the software lifecycle.



Each offering contributes a different capability while complementing existing CNAPP and CSPM investments.

CleanSight

Software Supply Chain Visibility

CNAPP platforms help organizations understand what is running in production. CleanSight extends that visibility by helping teams understand where software originated, how it wasassembled, and how it entered the environment.

By correlating container images with software dependencies, build pipelines, SBOMs, software provenance, and image inventory, CleanSight provides a continuously updated view of the software supply chain. It also helps organizations distinguish active software from unused, duplicate, and legacy images, allowing remediation efforts to focus on software that actually supports production workloads.

Clean Images

Verified Container Foundations

Clean Images help organizations reduce inherited software risk before applications are deployed.

Built directly from upstream source, they minimize unnecessary software, reduce attack surface, and provide verified container images with complete software provenance. Rather than replacing existing vulnerability management or runtime security tools, they improve the quality of the software entering production.

Clean Libraries

Verified Open Source Dependencies

Modern applications depend as much on open source libraries as they do on container images.

Libraries help engineering teams discover, evaluate, and consume verified open source libraries before those dependencies become part of an application. By introducing dependency governance earlier in the software lifecycle, Clean Libraries help organizations establish greater trust in the open source libraries developers and AI coding assistants consume before those dependencies enter the build process.

Working Together

Together, CleanSight, Clean Images, and Clean Libraries strengthen software before it reaches production by combining software supply chain visibility, verified software artifacts, and dependency governance. Rather than replacing existing CNAPP or CSPM investments, they complement them by addressing the stages of the software lifecycle that occur before runtime.

The CleanStart Difference
From Visibility to Verified Software

The previous sections highlighted an important distinction in software supply chain security. Visibility into software risk is essential, but visibility alone does not reduce that risk. Organizations also need confidence in the software they build with, deploy, and operate. That requires moving beyond identifying vulnerabilities toward verifying software before it reaches production.

This is the philosophy behind CleanStart. Rather than focusing exclusively on detecting vulnerabilities after software has been built, CleanStart helps organizations strengthen the software supply chain through visibility, verification, and governance across the earlier stages of the software lifecycle.

Visibility That Leads to Action

Security teams need more than a list of vulnerabilities. They need the context required to understand where those vulnerabilities originated, which software artifacts are affected, who owns them, and how they should be addressed.

CleanSight extends traditional runtime visibility with software supply chain visibility. By correlating software artifacts with their dependencies, provenance, ownership, and software inventory, it enables engineering and security teams to investigate the root cause of software risk instead of treating every vulnerability as an isolated finding. The result is more informed prioritization, faster remediation decisions, and greater confidence in the software entering production.

Verified Software Instead of Inherited Software

Most applications inherit thousands of software components before a single line of application code is added. Every inherited package becomes part of the software an organization is responsible for maintaining and securing.

Clean Images take a different approach by providing verified container images built directly from upstream source with a minimal software footprint and complete software provenance. Rather than accepting unnecessary inherited software as a default, organizations start from verified software foundations that reduce attack surface and minimize inherited risk before deployment.

The objective is straightforward: improve the quality of the software entering production so that security teams spend less time responding to vulnerabilities that never needed to exist in the first place.

Establishing Trust Before Software Is Consumed

Container images represent only one part of the software supply chain. Open source libraries introduce another significant source of inherited risk.

Libraries help organizations govern the libraries developers and AI coding assistants consume before those dependencies become part of an application. Instead of evaluating software only after it has entered the build process, organizations can establish trust earlier by verifying the software they choose to build with. This shifts dependency governance from a reactive activity to a proactive software supply chain practice.

Complementing Existing Security Investments

CleanStart is designed to complement existing CNAPP, CSPM, and runtime security investments rather than replace them.

By improving the software entering production, organizations enable their existing security tools to focus on operational threats instead of excessive inherited software risk. Fewer vulnerable components, stronger software provenance, and verified software artifacts lead to lower alert volumes, more effective prioritization, and a stronger overall software supply chain posture without requiring changes to existing security workflows.

Enterprise-Ready by Design

Enterprise software supply chain security requires more than technology. It also requires operational discipline, predictable remediation processes, and evidence that organizations can demonstrate during audits.

CleanStart supports these requirements through verified software artifacts, software provenance, signed attestations, and enterprise support backed by contractual service commitments for vulnerability remediation. Together, these capabilities help organizations strengthen their software supply chain while supporting regulatory and compliance requirements such as FedRAMP, PCI DSS v4.0, SOC 2 Type II, DORA, and NIS2.

Mapping the Two Layers of Modern Container Security

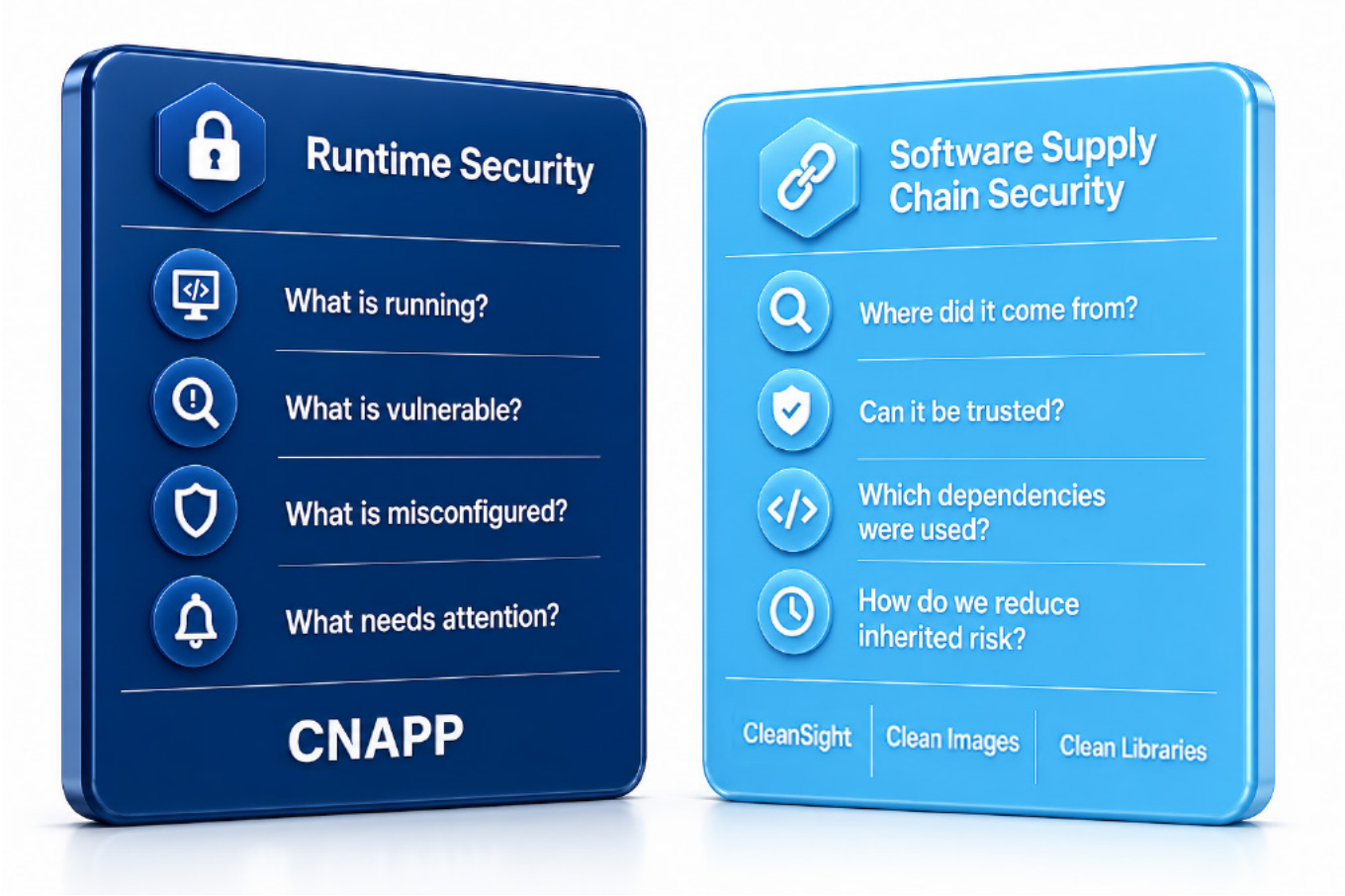
Modern container security spans two complementary layers. While both contribute to an organization's security posture, they answer fundamentally different questions and operate at different stages of the software lifecycle.

The runtime security layer focuses on software that has already been deployed. This is where CNAPP platforms provide visibility into workloads, cloud infrastructure, Kubernetes environments, and runtime behavior. Their purpose is to help security teams understand operational risk and respond to issues in production.

The software supply chain layer focuses on the software before it reaches production. This is where CleanSight, Clean Images, and Clean Libraries help organizations understand where software originated, verify the software entering the build process, and reduce inherited risk before applications are deployed.

Rather than viewing these capabilities as competing approaches, organizations should view them as complementary. Together, they provide answers to the questions security and engineering teams need to ask throughout the software lifecycle.

Modern Container Security



Organizations need both perspectives. Visibility helps security teams understand the state of production environments. Verification helps engineering teams improve the software those environments ultimately run. Together, they create a stronger and more resilient software supply chain.

Building an Audit-Ready Evidence Chain

For organizations operating under frameworks such as FedRAMP, PCI DSS v4.0, HIPAA, NIS2, DORA, and NIST, identifying vulnerabilities is only one part of a mature software security program. Organizations must also demonstrate how software is identified, verified, tracked, and remediated through a documented and repeatable process.Runtime scan results and alert history provide valuable operational evidence, but they represent only one part of that process. A comprehensive software supply chain security program also requires evidence of how software was built, what it contains, where it originated, and how vulnerabilities are managed throughout the software lifecycle.

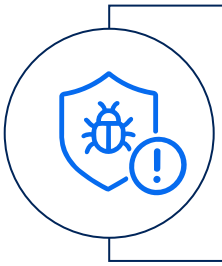
An audit-ready evidence chain typically includes:

Software Bills of Materials (SBOMs)

That document the software components included in every container image.

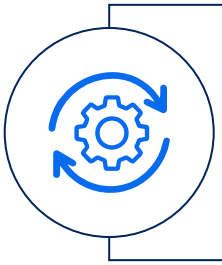
Software provenance and signed attestations

That demonstrate how software was built and verify its origin.



Vulnerability Exploitability eXchange (VEX) statements

That provide additional context about whether identified vulnerabilities are exploitable in a specific software configuration.



Documented remediation processes and service commitments

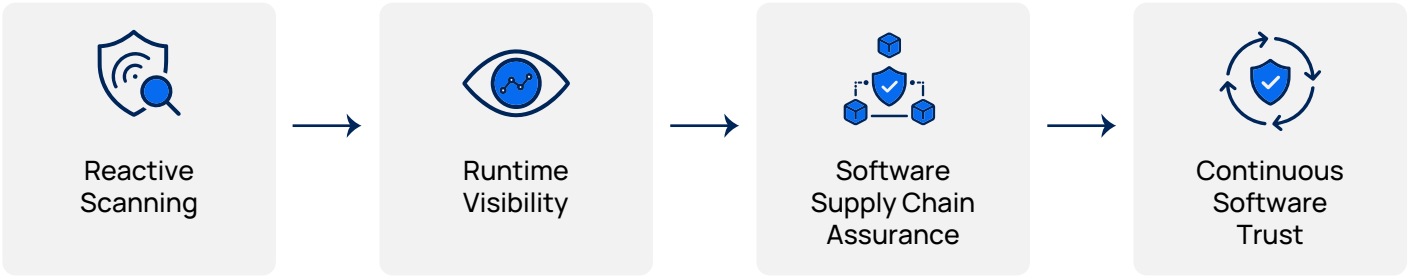
That demonstrate how newly disclosed vulnerabilities are evaluated and addressed.

Together, these artifacts provide an evidence chain that extends beyond vulnerability detection to include software provenance, verification, and documented remediation.

Within this evidence chain, each capability contributes different evidence. CNAPP platforms provide runtime visibility and operational findings. CleanSight contributes software supply chain visibility, inventory intelligence, and software provenance. Clean Images and Clean Libraries contribute verified software artifacts, signed attestations, and software foundations that strengthen an auditable software supply chain.

The Maturity Path
From Reactive Scanning to Verified Software

Most organizations progress through a series of maturity stages as they strengthen their software supply chain security. Each stage builds on the previous one, expanding visibility, improving software quality, and reducing inherited risk before applications reach production.



Stage 1: Reactive Vulnerability Scanning

Container images are scanned during development or deployment, and engineering teams manually triage findings based on severity. Over time, vulnerability backlogs grow faster than they can be remediated, while image registries accumulate unused, duplicate, and legacy images that continue to generate unnecessary alerts.

Primary Outcome: Visibility into vulnerabilities.

Stage 2: Runtime Visibility and Posture Management

Organizations introduce a CNAPP to improve visibility across cloud workloads, Kubernetes environments, identities, and runtime behavior. Risk prioritization improves, operational context becomes clearer, and security teams gain greater confidence in responding to production threats.

However, important questions remain unanswered. Teams still lack visibility into where software originated, which images should be retired, and how inherited software risk entered the environment in the first place.

Primary Outcome: Visibility into operational risk.

Stage 3: Software Supply Chain Assurance

Organizations extend their security program earlier in the software lifecycle by introducing software supply chain visibility, verified software artifacts, and dependency governance. Instead of relying solely on runtime detection, they begin improving the quality of the software entering production. As software quality improves upstream, inherited vulnerabilities decline and existing runtime security investments become more effective.

Primary Outcome: Reduced inherited software risk.

Stage 4:Continuous Software Trust

Security becomes a continuous software supply chain practice rather than a reactivevulnerability management exercise.

Verified software artifacts, software provenance, dependency governance, and enterprise remediation commitments work together to help organizations maintain stronger software foundations over time. Runtime security remains essential, but it operates on software that has already been strengthened before deployment.

The result is a software supply chain that is continuously verified, easier to govern, and better prepared to support security, compliance, and operational objectives.

Primary Outcome: Continuous software trust.

Conclusion
Complete the Journey

Visibility is an essential part of modern software security. But visibility alone does not strengthen the software entering production.

CNAPP and CSPM platforms have become indispensable for understanding runtime risk, monitoring cloud-native environments, and helping security teams respond to operational threats. They provide the visibility organizations need once software has been deployed.

Software supply chain security begins earlier.

It begins with understanding where software originated, verifying the container images and open source libraries entering the build process, and reducing inherited risk before applications ever reach production. It continues with software provenance, governance, and the evidence needed to demonstrate trust throughout the software lifecycle.

This is where CleanStart complements existing security investments.

CleanSight provides the software supply chain visibility needed to understand what software is running, where it came from, and how it entered the environment. Clean Images and Clean Libraries help organizations build on verified software artifacts, reducing inherited software risk before deployment while enabling existing runtime security tools to operate more effectively.

Modern container security is not about choosing between runtime visibility and software supply chain security. Organizations need both. One helps security teams understand the software they run. The other helps them improve the software they choose to run.

Because software security doesn't begin with the first alert.

It begins with the software you build, the software you consume, and the software you choose to trust.

Ready to strengthen your software supply chain?

Contact the CleanStart team to learn how CleanSight, Clean Images, and Clean Libraries can complement your existing cloud security investments.

Version 1.5.7 I July 2026