WHITEPAPER

# What We Run but Cannot See: Rethinking Trust in Software

CLAIRVOYANT

# Contents

WHITEPAPER

# What We Run but Cannot See: Rethinking Trust in Software

Recent developments across the industry are pointing to a structural gap in how organizations secure modern environments.

The latest findings from the M-Trends Report show that attackers are able to operate inside enterprise environments for extended periods, often using legitimate tools and credentials without triggering alerts. In many cases, activity blends into normal operations, making it difficult to distinguish between expected behaviour and malicious intent.

Researchers have also demonstrated a supply chain attack technique using "invisible" code embedded in repositories, where malicious instructions can be hidden from human reviewers while still being executed by systems (as reported by Ars Technica report on invisible code supply chain attack). In practical terms, code can appear clean during review, yet behave differently once it runs.

More recently, an incident involving Anthropic's Claude exposed internal system logic and prompt structures. This provided insight into how the model operates and how its safeguards are implemented. While not a traditional breach, it demonstrated that critical application logic can be exposed and potentially understood or manipulated in unintended ways.

Taken together, these examples reflect a broader shift in how software is created, distributed, and trusted across modern environments.

# Why This Matters Now

The impact is already visible across operational and regulatory environments. Across sectors such as financial services, telecommunications, and government, software has become the primary interface through which systems operate. The volume, velocity, and origin of software entering these environments continue to increase.

Regulatory expectations are evolving. In Singapore, for example, the Cyber Security Agency of Singapore has introduced enhanced requirements for Critical Information Infrastructure operators and their vendors, including the adoption of the Cyber Trust Mark framework. These measures extend beyond internal systems to include third-party providers and supporting infrastructure.

*This reflects a broader shift. Organizations are increasingly expected to demonstrate not only compliance, but confidence in the systems and software they rely on, including those they do not directly build or control.*

At the same time, development practices are changing. Software is assembled from multiple sources, updated continuously, and in many cases generated or assisted by AI. The level of implicit trust placed on software has expanded, while the ability to independently validate that trust has not kept pace.

Organizations today rely on a mix of open-source components, third-party applications, internally developed tools, and increasingly, AI-generated code. Software is introduced continuously through updates, integrations, and decentralized development practices.

In most environments, visibility into software is limited to origin and metadata. Teams can identify where software comes from, whether it is signed, and which components are included. There is limited visibility into how that software behaves once it is compiled and deployed.

Security controls have evolved to address known risks. Vulnerability management identifies known weaknesses. Endpoint and network controls focus on detecting malicious activity. However, these controls operate after software has already been introduced into the environment.

Incident response continues to show that attackers operate within normal system boundaries, using legitimate tools and credentials. In these situations, detection signals are often limited or delayed.

This creates a gap between what organizations deploy and what they can independently verify.

# Expanding Software Risk Exposure

This gap becomes more pronounced as organizations rely on software from a broader range of sources.

This includes third-party applications, internally developed software, and code generated or introduced through modern development and AI-assisted workflows.

In each of these scenarios, software is trusted and deployed based on origin, reputation, or development process.

However, these factors do not provide a complete understanding of software risk.

## Key scenarios include:

**Third-party software**

Software from vendors or external sources is trusted based on reputation, certification, or use, without an independent assessment of its behaviour.

**Internally developed software**

Applications built in-house are validated through development and testing processes, but are not always independently verified beyond their intended functionality.

**AI-generated or AI-assisted code**

Code produced through AI tools accelerates development, but introduces logic that may not be fully reviewed, explainable, or validated.

**Shadow IT software**

Tools and applications adopted outside centralized governance may enter the environment without formal validation or oversight.

Across these use cases, the challenge is consistent.

Software is introduced into production environments without an independent assessment of how it behaves.

This creates a growing gap between the level of trust placed on software and the level of assurance that actually exists.

# Reframing the Security Model

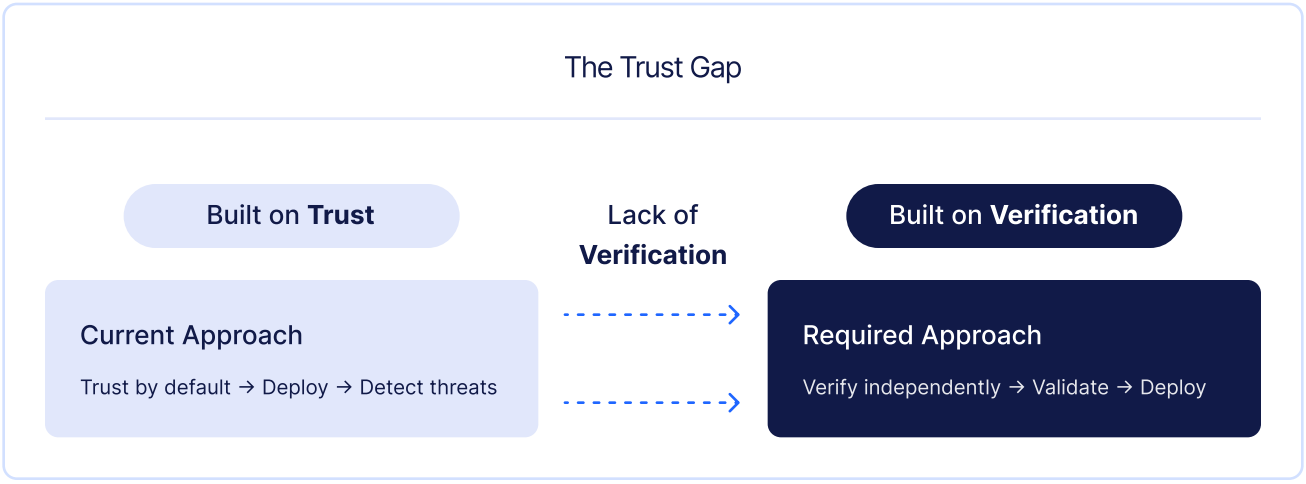
The current security model assumes that risk will eventually manifest as observable activity. In many environments, this assumption does not consistently hold.

Effective attacks increasingly operate through legitimate processes, trusted software, and expected system behaviour. As a result, detection may be delayed, incomplete, or dependent on downstream impact.

This reflects a limitation in how control points are currently applied.

> *When software is trusted by default and its behaviour is not independently examined, that trust is inherited by the environment without validation.*

A more resilient approach requires introducing validation earlier in the lifecycle, with greater emphasis on understanding what is allowed to execute before it enters production.



# Evolving Approaches to Software Security

Over the past decade, the industry has made significant progress in improving software security.

A range of solutions now exist to help organizations:

- Identify vulnerabilities in code
- Manage open-source dependencies
- Improve secure development practices
- Strengthen software supply chain governance

These approaches, including developer-focused and shift-left security models, are an important part of modern security programs.

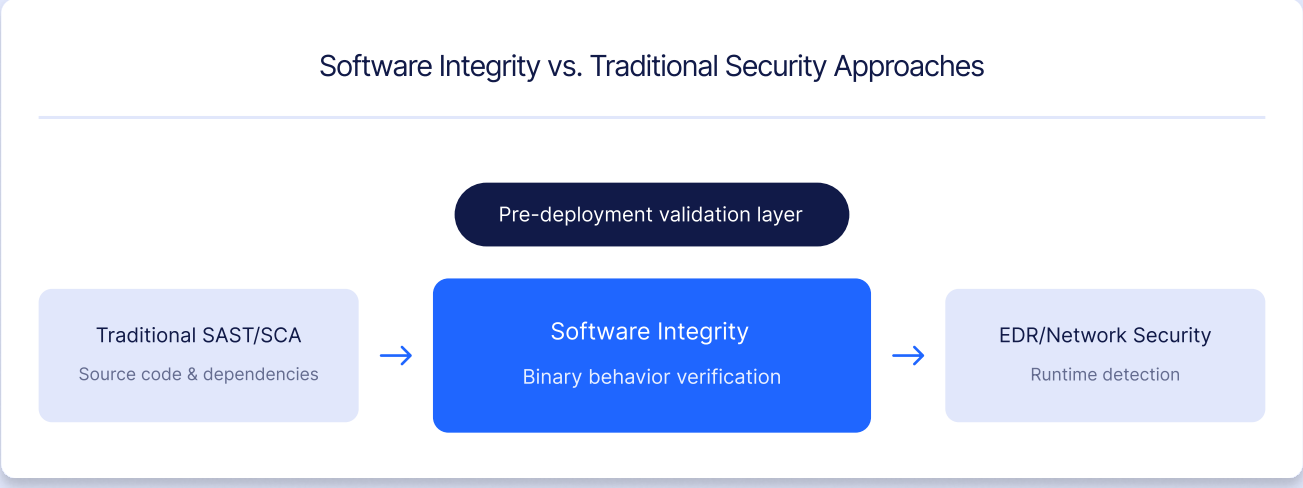
They primarily address how software is written, maintained, and sourced.

However, they do not fully address a separate requirement that is becoming more visible: understanding how software behaves once it is compiled and introduced into the environment.

# A Different Approach

A different approach is emerging to address this gap.  Rather than focusing on source code, dependencies, or development practices, this approach centers on establishing software integrity through independent verification.

Clairvoyant's Software Integrity Platform enables organizations to gain visibility into software risk and establish trust in software before it is introduced into the environment.




This allows organizations to:

- Identify risks that are not visible through traditional application security or supply chain tools
- Understand how software behaves beyond its declared functionality
- Validate alignment between intended purpose and observed behaviour
- Establish confidence in software prior to deployment

This approach applies across third-party software, internally developed applications, and software introduced through automated or decentralized processes, including AI-assisted development.

# Operational and Compliance Impact

In many organizations, assurance of software — particularly for third-party or critical systems — is a manual and time-intensive process.

Security and GRC teams rely on vendor attestations, documentation reviews, and, in some cases, deeper technical analysis to establish confidence. This process can take days or weeks and is difficult to scale across the volume of software being introduced.

As regulatory expectations increase, the burden of demonstrating assurance continues to grow. Introducing a structured approach to evaluating software prior to deployment improves this process.

## Organizations can:

Reduce the time required to evaluate and approve software

Improve consistency in assurance decisions

Support compliance requirements with more direct evidence

Enable faster deployment with greater confidence

This improves both security outcomes and operational efficiency.

# Industry Direction and Alignment

This shift is becoming more visible across both industry research and operational practice.

At the recent RSA Conference 2026, discussions increasingly focused on the limitations of existing trust models, particularly in environments where software is continuously introduced, updated, and executed without independent validation.

Research from firms such as Gartner has similarly emphasized the need to move beyond approaches centred on vulnerabilities and dependencies, toward stronger assurance of software itself.

At the same time, incident response findings continue to show that attackers operate through legitimate software, credentials, and system processes. Reports such as the Verizon Data Breach Investigations Report and the CrowdStrike Global Threat Report highlight how malicious activity increasingly blends into expected behaviour.

Converging Industry Signals

1. **Research & Conferences**
   Focus shifting from prevention to verification
2. **Threat Intelligence**
   Attackers using legitimate tools and processes
3. **Regulatory Evolution**
   Greater emphasis on software assurance

Taken together, these signals point to a consistent direction.

> Security is evolving from identifying malicious activity to establishing confidence in what is allowed to run.

# Conclusion

As software becomes the foundation of modern systems, the question is no longer whether threats can be detected.

The more immediate question is whether the software itself has been sufficiently understood and verified before it is allowed to run.

> *Organizations are increasingly operating in environments where software is trusted by default, but not independently validated.*

This gap is becoming harder to manage as software volume increases, development accelerates, and regulatory expectations continue to rise.

Closing this gap requires a shift. Not away from existing controls, but toward establishing confidence in software before it is introduced into the environment.

This is where software integrity becomes essential

---

# Next Steps

To learn more about how Clairvoyant can help your organization establish software integrity and independent verification:

- Request a demonstration of the Software Integrity Platform
- Discuss your organization's specific requirements
- Explore integration with your existing security infrastructure

# References


1. *Mandiant, M-Trends Report*
   https://www.mandiant.com/resources/m-trends

2. *Ars Technica, Supply chain attack using invisible code hits GitHub and other repositories*
   https://arstechnica.com/security/2026/03/supply-chain-attack-using-invisible-code-hits-github-and-other-repositories/

3. *CNBC, Anthropic leak reveals Claude internal system details*
   https://www.cnbc.com/2026/03/31/anthropic-leak-claude-code-internal-source.html

4. *Cyber Security Agency of Singapore (CSA), Cyber Trust Mark and CII security requirements*
   https://www.csa.gov.sg

5. *Verizon, Data Breach Investigations Report*
   https://www.verizon.com/business/resources/reports/dbir/

6. *CrowdStrike, Global Threat Report*
   https://www.crowdstrike.com/resources/reports/global-threat-report/

7. *RSA Conference 2026*
   https://www.rsaconference.com


## The Company

Founded by cybersecurity startup veterans from FireEye and Menlo Security, Clairvoyant Intelligence is a cybersecurity leader making a big impact in the area of software assurance at speed and scale.  Experience and past performance with US Civilian and Department of Air Force.