The AppSec Exit Plan

How to cheat on your scanner (and get away with it).

boost security.io

Is your relationship with AppSec scanners more like a hostage situation?

If you're like most security leaders we talk to about replacing legacy SCA and SAST tools like Snyk and Veracode, you're probably not loyal to your tools. Maybe you're neutral, maybe they make you feel anything from mild irritation to outright annoyance...but even if you're not happy with them, you probably feel stuck.

Does this sound like you?

> **You don't really love your scanning tools,** but you're worried that if you switch for something else, you'll end up with the same problems you have now.

> **Vendor costs keep rising,** but you're not getting real additional value for the amount you're paying. You have a harder and harder time justifying your renewal costs to the CFO or board.

> **Your SCA and SAST were built for a slower SDLC,** and agentic coding makes your carefully cultivated security program feel like it's held together with duct tape.

> **You've caught yourself wondering what else is out there,** then talked yourself out of looking. Deployment was hard enough the first time. If the next scanner takes as long to deploy as this one, a year-long rip-and-replace project sounds about as fun as chewing glass. (Spoiler alert: it doesn't have to be this way.)

We're Boost, and we know how these vendors operate because we used to work for them. We built our product to design an easier, faster, better way out than anything else on the market.

This guide shows you how you can see what you've been missing, without a migration project or a permission slip from engineering.

Shhhh...setting up your discreet rendezvous with Boost in Silent Mode

A rip-and-replace needs time and political capital you don't have. We've been there and done that (and have a closet full of Black Hat t-shirts), so we built our product with real security teams and their real-life constraints in mind.

Boost's Zero-Touch Provisioning makes deployment fast and so much easier than legacy AppSec tools like Snyk, Veracode, Mend, and others. You won't need engineering to say "yes." You also won't need to flood your devs with notifications. We're the easiest deployment in the business, and we mean it.

Easy as 1-2-3

1

Hand us a key.

Create a read-only service account in your source control manager and generate an API key. That key is the whole integration. Nothing gets inserted into a pipeline, and your code never leaves your environment.

2

We map everything.

Boost connects at the source control level and sees every repo you've got, including the ones your current scanner never knew existed. New repos get picked up automatically as your developers create them.

3

You watch in Silent Mode.

Findings go to a dashboard only you see, so a new tool never blows up your developers' notifications. Boost scans every pull request and every merge, building your side-by-side while your legacy scanner carries on none the wiser.

Side-by-side: seeing clearly for the first time in a long time

When you run Boost in Silent Mode next to your legacy scanner, you'll find out a lot about what you've been putting up with (and what you've been missing).

01 The fixes actually get fixed

Your old scanner files a ticket and the finding rots in a backlog nobody clears. Boost puts the fix right in the pull request, where developers already are, worded so they can act on it without a meeting.

> *One AppSec lead watched her team log 530 fixes in two weeks, clearing a backlog years in the making. Her verdict: developers actually fix things with Boost, and she'd guarantee that wouldn't happen with another tool.*

02 Keep up with your own developers

Your developers already work with agents. They're generating code, pulling dependencies, and wiring up MCP servers every day, and the models are getting good (maybe scary good). Your scanner was built for a slower world and can't follow them into this one. Oh yeah: and the bad guys have access to the same coding agents your devs do.

Boost was built for exactly this shift. It runs where the work now happens, on the laptop, and shows you every coding agent, MCP server, and IDE extension your developers are actually using. With added features from Boost Developer Endpoint Protection, the Boost Platform can stop a hallucinated package the moment an agent reaches for it, and catch credentials leaking out of a prompt before they reach someone else's model.

03 Finally set the rules

Your old scanner decides what counts as a finding, and you live with its guesses. Boost hands you the controls.

> *One security leader picked Boost over GitLab Ultimate and a few others because she could finally write rules precise enough to separate a theoretical flaw from a real one. She built a check that enforced repository ownership tags, then applied it across thousands of repos without touching a pipeline, because you can't route a fix when nobody knows who owns the code.*

04 See your whole estate

Your old scanner watches the repos you pointed it at years ago. Everything since is a blind spot.

> *Another lead inherited tools that left her guessing at her own attack surface. "I couldn't see what was active," she said. Boost mapped her real footprint the day she connected it, live repos and dead ones, and she measured an 80% jump in how much of her surface she could actually see.*

05 Enjoy the silence

Your old scanner alerts on every theoretical CVE and buries the real ones. Boost checks whether a flaw sits in code that ever actually runs, and clears the rest.

Once the findings are trustworthy, you get to do what a noisy scanner never let you: stop bad code from shipping instead of just warning a developer after the fact.

Side by Side: Boost vs. Legacy vendors

	Legacy scanner	Boost
Findings	Every theoretical CVE	**Only what's reachable at runtime**
Remediation	A Jira ticket	**A fix in the PR, ready to merge**
New repos	Added by hand, if someone remembers	**Detected automatically**
The laptop / AI agents	Invisible	**Inventoried and protected**
Deployment	Pipeline edits, developer favors	**One API key, no pipeline changes**
Policy	The vendor's severity guesses	**Your rules, enforced everywhere**

What's so great about Boost?

We've spent this guide on where legacy scanners fall short. Here's everything Boost does, in one place.

ASPM & Risk Prioritization

Your SAST and SCA in one place, with reachability analysis that separates material risk from theoretical noise. A granular policy engine you configure to your own definition of risk. Automatic discovery of every repo, active and archived.

Developer Endpoint Protection

Inventory of every coding agent, local model, and IDE extension on the machine. MCP server validation and governance. Prompt sanitization that masks credentials before they leave the laptop. Local secret exposure detection in dotfiles and environment variables.

AI Agent Governance

Secure coding standards injected into the agent's context before it writes a line. A continuous AI-BOM tracking every model, agent, and permission touching your code. Policy enforced at the point of generation.

Software Supply Chain Security

The SCA you're shopping for, plus the parts it misses: pre-ingestion blocking that stops a typosquatted or hallucinated package before it ever downloads. Dependency and container scanning. CI/CD pipeline integrity against living-off-the-pipeline attacks.

Remediation & Execution

Contextual fixes generated and pushed straight into the pull request. One-click merge. A defensible audit trail behind every automated change.

Deployment & Scale

Zero-Touch Provisioning through a single service account. No pipeline edits or developer sign-off needed. Continuous scanning of every PR and merge across thousands of repos.

Be honest with yourself:

Are you staying with your AppSec tool because you love it...

Or because you dread the thought of leaving and starting over?

If you're still reading this guide, it's because **you already know the answer.**

We've been in your shoes before. We built Boost because **we truly believe you deserve better.**

We know change isn't easy, and that's why we created Silent Mode: to give you the chance to see Boost's real findings in your real repos, before devs ever find out.

> **You don't have to keep making excuses for tools you never loved anyway. Book a call with our team, and we'll set up Silent Mode together so you can see what your repos have to say.**

boost security.io