AppSec Tool Evaluation Scorecard

A practical decision framework for development managers, DevSecOps leads & CISOs · Based on *Mastering the Art of Application Security Testing* by Julian Totzek-Hallhuber

HOW TO USE: *For each criterion, score your candidate tool 1–4 using the descriptions below. Weight criteria by importance to your context. Total score out of 68 (SAST 20 · DAST 16 · SCA 16 · Container/IaC 16).* ***Score ≥50: strong candidate · 35–49: viable with gaps · <35: reconsider.***

SAST — Static Application Security Testing

Score →

Evaluation Criterion	What to assess	1 — Poor	2 — Adequate	3 — Good	4 — Excellent
Language & framework coverage	Does it support your full tech stack (languages, frameworks, build systems)?	<3 languages	3–8 languages	8–15 languages	Full stack coverage
False positive rate	Measured in YOUR codebase, not vendor demo. Below 10% is acceptable.	30%+ FP rate	10–20% FP rate	5%–10% FP rate	<5% FP rate
Fix guidance quality	Does output give actionable remediation advice developers will actually use?	Location only	Rule description	Code examples	Step-by-step fix + context
CI/CD integration speed	Scan time in your pipeline. Slow tools get switched off.	>45 min	15–45 min	5–15 min	<5 min
Noise management	Suppression, waivers, accepted-risk workflows. Can findings be triaged at scale?	No workflow	Manual only	Basic automation	Full triage + policy engine

DAST — Dynamic Application Security Testing

Score →

Evaluation Criterion	What to assess	1 — Poor	2 — Adequate	3 — Good	4 — Excellent
Authentication support	Can the scanner authenticate and reach protected parts of the application?	Unauthenticated only	Basic auth	Form/SSO login	Full OAuth/API token
API coverage	REST, GraphQL, SOAP — does it cover your API surface, not just web UI?	UI-only	REST basic	REST + OpenAPI	REST + GraphQL + SOAP
Scan speed vs. coverage	Full crawl time in staging environment. Balance depth with pipeline fit.	>4 hours	2–4 hours	30 min–2 hrs	<30 min (targeted)
Finding deduplication	Does it filter duplicate findings, or will your team drown in repeated alerts?	No dedup	Basic URL dedup	Parameter-level dedup	Intelligent clustering

SCA — Software Composition Analysis

Score →

Evaluation Criterion	What to assess	1 — Poor	2 — Adequate	3 — Good	4 — Excellent
Vulnerability database freshness	How quickly are new CVEs reflected? Hours vs. days matters post-disclosure.	>7 days lag	2–7 days lag	Same-day updates	Real-time / near real-time
Reachability analysis	Can it determine if a vulnerable dependency is actually called in your code?	No reachability	Package-level only	Call graph basic	Full reachability
License compliance	Automated detection of copyleft/commercial-risk licences in the dependency tree.	Not included	Direct deps only	Transitive deps	Policy enforcement
Fix recommendations	Does it suggest the minimum safe version upgrade, not just flag the vulnerability?	Flag only	Latest version	Min safe version	PR auto-generation

Container & IaC Security Scanning

Score →

Evaluation Criterion	What to assess	1 — Poor	2 — Adequate	3 — Good	4 — Excellent
Container image scanning depth	OS packages, application layers, secrets in layers — not just base image CVEs.	Base image only	+ App packages	+ Secrets detection	Full layer analysis
IaC framework coverage	Terraform, CloudFormation, Kubernetes manifests, Helm, Bicep — what's supported?	1 framework	2–3 frameworks	4–5 frameworks	Full IaC stack
Shift-left integration	Can developers run checks locally (IDE plugin, pre-commit hook) before pipeline?	CI/CD only	CLI available	IDE plugin	IDE + pre-commit + CI/CD
Misconfiguration vs. CVE balance	Does it detect both known vulnerabilities AND configuration best-practice violations?	CVEs only	CVE + basic config	CVE + CIS benchmarks	CVE + config + policy

SCORING SUMMARY — Compare vendors side by side

VENDOR	SAST /20	DAST /16	SCA /16	Container & IaC /16	TOTAL /68	GO / NO-GO
Vendor 1						
Vendor 2						
Vendor 3						