

It passed static analysis. It is still exploitable.

Test it the way an attacker will.

Xygeni DAST attacks your running web applications and APIs from the outside, in your own infrastructure, and hands developers the evidence and a proof of concept to reproduce each finding.

200+ active and passive detection rules, plus template-based checks against known CVEs and misconfigurations.

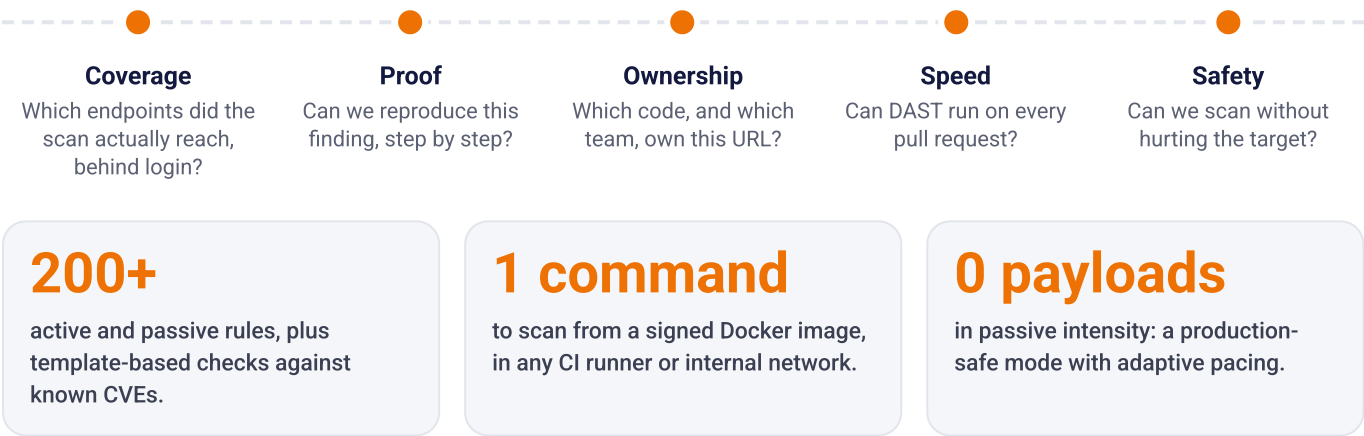
THE PROBLEM

Runtime risk lives where static analysis cannot see.

Code that passes every static check can still be exploitable once deployed: authentication flaws, access-control weaknesses, misconfigurations and injection paths that only appear at runtime. Traditional DAST runs rarely and outside CI, stalls at the login page, and reports a URL instead of the code that has to change. So findings get pushed back.

Xygeni DAST runs in your pipelines and infrastructure, scans behind modern authentication, re-tests only what changed, and ties each runtime finding to the code that serves it.

FIVE QUESTIONS MOST TEAMS CANNOT ANSWER TODAY



ONE PRODUCT, THREE STEPS

Discover. Attack. Fix.



Discover the real surface. **Attack** it safely. **Fix** it in the code.

Without Xygeni	With Xygeni
DAST runs rarely, outside CI	Incremental scans on every pull request
Scans stall at the login page	Authenticated scanning, including SSO
A URL and a severity	HTTP evidence and a reproducible PoC
Runtime findings with no code owner	Correlated to the handler and its SAST findings

DISCOVER · COVERAGE

Reach the endpoints that matter, behind the login

Two-axis scan profiles, deep crawl, traffic seeding and modern authentication.

Xygeni DAST combines a spider, an AJAX browser crawl and an optional headless deep crawl, and seeds the scan from API definitions and recorded traffic, so endpoints a crawler would miss are still tested. Scan profiles adapt to what the target is, and to how hard it should be tested.

TWO-AXIS SCAN PROFILES		TECH BASE →					
		Traditional	SPA	OpenAPI	GraphQL	SOAP	CMS
Quick fast CI gate		☐	☐	☐	☐	☐	☐
Balanced default		☐	☐	☐	☐	☐	☐
Deep max strength, CVE check		☐	☐	☒	☐	☐	☐
Passive no attack payloads		☐	☐	☐	☐	☐	☐

--profile openapi --intensity deep

Schematic: scan profiles combine a tech base with an intensity, auto-selected or defined as YAML in the repository.

```
# nightly, full scan
$ xy-dast scan -u https://staging.example.com \
  --profile spa --intensity balanced \
  --deep-crawl --navigation login.side \
  --fail-on high

# on every pull request
$ xy-dast scan -u https://staging.example.com \
  --incremental --intensity quick \
  --fail-on high
```

Schematic: a full scan nightly and an incremental scan on every pull request, gated by severity.

- **Application types:** server-rendered apps, SPAs, REST via OpenAPI or Postman, GraphQL, SOAP and CMS sites.
- **Deep crawl:** a headless, JavaScript-aware crawler seeds URLs before the main scan.
- **Authentication:** form, bearer, API key, basic, mTLS and OAuth2, with token refresh for long scans.
- **Two-axis profiles:** tech base plus intensity (quick, balanced, deep or passive), auto-selected or custom.
- **Traffic seeding:** Selenium IDE and Chrome DevTools recordings replayed, HAR files imported.
- **SSO:** pre-authenticated session import, including Playwright storage state, and scripted browser logins.

Guided setup wizard

Signed Docker image

amd64 and arm64

Secrets as environment references

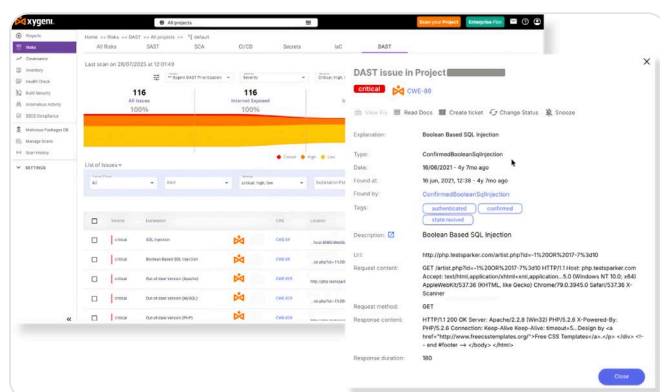
For the AppSec lead: coverage of the real application, not just the login page.

ATTACK · RUNTIME PROOF

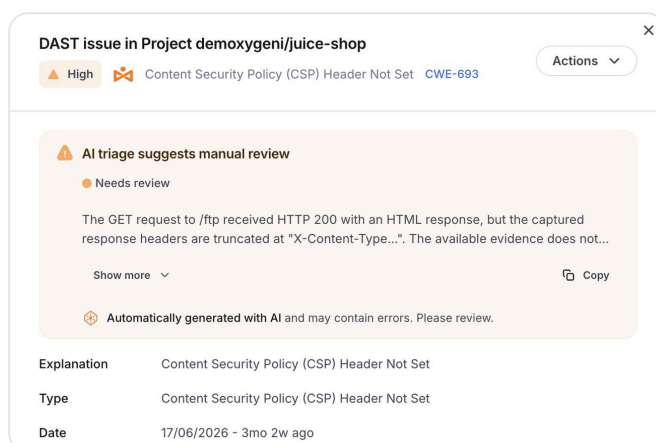
Prove what is exploitable, with evidence developers can replay

Active and passive rules, CVE checks and out-of-band detection. Safe by design.

Each scan simulates real attacks against live endpoints: injection, XSS, authentication and session flaws, access-control weaknesses, misconfigurations and known CVEs. Every finding carries the full HTTP request and response and a potential proof of concept, so developers can reproduce it instead of debating it.



A DAST finding with its HTTP request, response and evidence, ready for the developer.



AI Triage on a DAST finding: when the evidence is ambiguous, the verdict is needs review.

- **200+ detectors:** SQL and command injection, XSS, CSRF, session fixation, path traversal, IDOR, SSTI and more.
- **Out-of-band detection:** blind SSRF, blind XXE, Log4Shell and blind SSTI, with a self-hosted callback option.
- **Safe operation:** rate limits, adaptive pacing on 429 and 503 responses, and a passive mode with no attack payloads.
- **Known CVEs:** template-based checks of discovered endpoints against thousands of CVEs and exposures.
- **Potential PoC:** a curl command plus the expected response, with a confidence level.
- **Traffic tagging:** every request is marked, so application owners can identify scan traffic.

CWE

OWASP Top 10 2025

OWASP API Top 10

PCI DSS 4.0

NIST 800-53

A throttled scan is never reported as a clean one. Pacing is recorded in the report, and in incremental runs unchanged endpoints keep their previous findings, marked kept-untested, instead of silently passing.

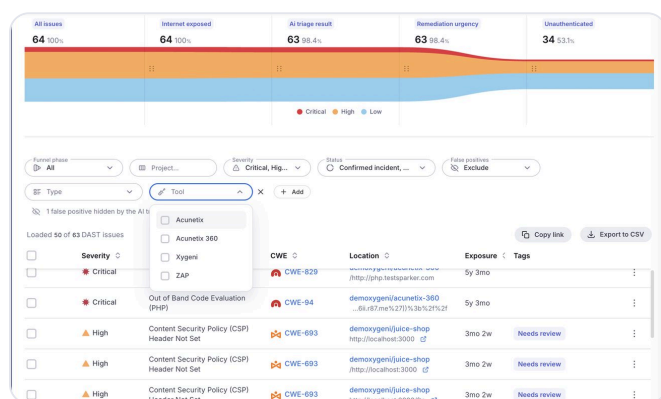
For developers: a finding that comes with the request to replay it.

FIX · IN THE WORKFLOW

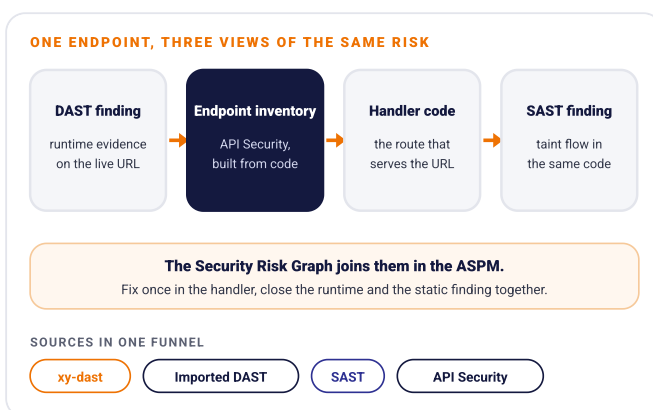
DAST in the pull request, tied to the code that serves it

Incremental scanning, correlation with SAST and API Security, AI Triage and one workflow.

Incremental DAST re-tests only the endpoints whose code changed, using API Security's semantic endpoint signatures or a diff of your API definitions, and merges the results into a full snapshot. Findings land in the ASPM, where the endpoint inventory joins a runtime finding to the handler that serves it and to the SAST findings in that code.



The DAST funnel with AI stages, and findings from Xygeni and other DAST tools in one view.



Schematic: the endpoint inventory joins runtime and static findings for the same route.

- **Incremental by design:** changed endpoints get fresh results; broad changes fall back to a full scan.
- **CI/CD gates:** fail on a severity threshold, SARIF output for code scanning, GitHub Actions and GitLab CI examples.
- **AI Triage:** verdict, urgency and complexity from the HTTP evidence.
- **Recommended rhythm:** incremental on pull requests, full scans nightly or on merge.
- **Correlation:** DAST, SAST and API Security findings joined through one endpoint inventory.
- **Keep your current DAST:** results from other DAST tools imported into the same funnel and workflow.

API Security

Builds the endpoint inventory from code, so DAST tests endpoints a crawl never reaches.

AI Pentesting

Goes one step further and proves exploitability with a deterministic oracle, including authorization flaws.

ASPM

Runtime findings next to code, dependency, secrets and IaC risk in one model.

For VP Engineering: runtime testing that fits the release rhythm instead of blocking it.

WHY XYGENI FOR DAST

Discover the real surface. Attack it safely. Fix it in the code.

Xygeni DAST tests running applications and APIs like an attacker, inside your own infrastructure, and connects every runtime finding to the code and the team that can fix it.

Runtime evidence

Full HTTP request and response and a potential PoC on every finding.

DAST on every PR

Incremental scans re-test only what changed, with honest kept-untested results.

From URL to code

Runtime findings correlated with API Security and SAST through one endpoint inventory.

Runs in your infrastructure

Signed Docker image, internal targets, production-safe passive mode.

Why security leaders choose Xygeni

Proven in production

Banco Serfinanza reports around 90% less time identifying, prioritizing and assigning findings; Fintonic, a 90% reduction in prioritization time.

One platform

Code, open source, secrets, IaC, pipelines, running apps and AI in one inventory, one risk model and one workflow.

Extend, don't replace

Xygeni's AI layer triages, explains and fixes findings from the security tools you already run.

Built for regulated environments

European vendor. DAST runs inside your own infrastructure; the platform is SaaS or on-premise, with any LLM.



Runtime security that starts where attacks begin, and ends in the code

Discover. Attack. Fix. One platform.

[Request a PoC](#)

[Book a demo](#)

- Keep your current DAST: Xygeni ingests its findings
- Runs in your CI runners and internal networks
- xygeni.io

RECOGNIZED IN 2026



Hot Company in Application Security Posture Management 2026

Global InfoSec Awards · Cyber Defense Magazine