perplexity

CAST Highlight: Features, Inner Workings & How to Re-Create Them with Checkmarx, Black Duck, TruffleHog, and SonarQube

Overview
CAST Highlight is a SaaS “software intelligence” platform that performs ultra-rapid source-code scanning to give executives and engineers portfolio-level insight into cloud-readiness, technical debt, software health, open-source risk, security, and environmental impact[1] [2]. This report (1) enumerates every major capability of CAST Highlight, detailing the underlying mechanics, and (2) shows how to assemble an equivalent capability stack by combining Checkmarx, Synopsys Black Duck, TruffleHog, and SonarQube. A final blueprint maps each CAST Highlight feature to the alternative tools and explains technical implementation patterns, data flows, and integration tips.

Table of Contents

- Cloud-Readiness Analytics
- Software Composition Analysis & Open-Source Governance
- Technical Debt & Software Health
- Security Weakness Prediction (Pre-CVE)
- Green Impact Scoring
- Portfolio Management & Business Context Surveys
- Rapid, Non-Intrusive Scanning Architecture
- Detailed Feature-by-Feature Comparison Table
- Building a “CAST Highlight-like” Stack with Checkmarx, Black Duck, TruffleHog & SonarQube
- Integration Blueprints & Data Pipelines
- Governance, Reporting & Executive Dashboards
- Limitations, Edge Cases, and Future Enhancements

Cloud-Readiness Analytics

1. Cloud Ready Index

CAST Highlight parses all source files (40+ languages) and applies pattern-matching rules to label *Blockers* (e.g., raw file system access, hard-coded IPs) and *Boosters* (cloud-friendly constructs such as RESTful services)[3] [4]. Scores are rolled into a 0–100 Cloud Ready Index per application and aggregated across the portfolio.

2. Effort & Wave Planning

Each blocker carries a default remediation effort (developer-days) calibrated by CAST's benchmarking data; total effort yields a migration *temperature map* that groups apps into *Rehost, Refactor, Rearchitect, Rebuild*, or *Retire* waves[5] [6].

3. Cloud-Native Service Recommendations

Using language/technology fingerprints, Highlight proposes AWS, Azure, or GCP managed services that can replace on-prem equivalents (e.g., swap Oracle DB calls for Amazon RDS)[3] [6].

Software Composition Analysis (SCA) & Open-Source Governance

4. BOM Creation, License & Vulnerability Scanning

The local agent fingerprints every third-party file and manifest to build a complete SBOM, cross-checks 350+ license families, and flags risk levels (High = GPL-style reciprocal obligations requiring source disclosure)[7].

5. Portfolio Advisor for Open Source & "Sherpa" Recommender

This dashboard ranks applications by legal, security, or obsolescence exposure and auto-suggests safer component versions[8].

6. Emerging Weakness Detection

Highlight inspects popular OSS projects for CWEs even before a CVE is issued, allowing "vulnerability prediction" for future CVEs[9].

Technical Debt & Overall Software Health

7. Resiliency, Agility, Elegance & Complexity Metrics

CAST computes structural quality violations (such as cyclic dependencies or overly complex methods) and translates them into *Software Health* factors[1] [10].

8. Portfolio Advisor for Technical Debt

A sunburst visualization breaks total debt into contributing factors and highlights the 20% of applications representing 80% of remediation payoff[11] [10].

9. Monetary Valuation

Debt is monetized using CAST's empirically derived $3.61 per LOC benchmark[12].

Security Weakness Prediction (Pre-CVE)

Highlight extrapolates likely future CVEs by tracing CWEs inside libraries, covering ~67% of future known vulnerabilities[9].

Green Impact Scoring

CAST tallies energy-inefficient code patterns (excessive I/O, nested loops, etc.) and converts them into CO_2-equivalent emissions, supplying remediation hints[13].

Portfolio Management & Qualitative Context

Business surveys (criticality, revenue impact, user volumes) merge with code-level analytics, enabling risk-versus-value bubble charts for rationalization[1].

Rapid, Non-Intrusive Scanning Architecture

- **Local Agent**: Windows desktop binary; no source code leaves the premises—only encrypted CSV metrics[14].
- **Upload Step**: Results file posted over HTTPS/TLS-256 to the SaaS portal; dashboards refresh in minutes[3].
- **CI/CD Plugins & REST API**: Automate recurring scans and data extraction into BI tools[1].

Detailed Feature-by-Feature Comparison

Capability	CAST Highlight	Checkmarx	Black Duck	TruffleHog	SonarQube	Notes & Gaps
Cloud-Readiness scoring	Yes: Blocker/Booster rules, effort, 5 R roadmap[3] [4]	Roadmap plug-ins only (no built-in) [15]	No[16]	No[17]	No[18]	Custom scripts needed with alt-stack
SCA (license + vuln)	350+ licenses; BDSA-style insights[7] [8]	Integrated SCA module (Checkmarx One)[19] [20]	Core feature: KnowledgeBase, SBOM, BDSA[21] [22]	No	Limited (third-party plug-ins)	Combine Checkmarx SCA or Black Duck
Secrets detection	Limited (focus on code metrics)	Secrets Detection module[23]	No	Core mission: 800+ detectors, active validation[24] [25]	Possible via plug-in	Add TruffleHog
SAST (custom code vulns)	Basic OWASP Top 10 heuristics inside Software Health	Enterprise-grade SAST (75+ languages) [26] [23]	No (OSS only)	No	Built-in static analysis for bugs & smells[18] [27]	Combine Checkmarx + SonarQube

Capability	CAST Highlight	Checkmarx	Black Duck	TruffleHog	SonarQube	Notes & Gaps
Technical Debt monetization	Built-in $/LOC model; Portfolio Advisor [11] [12]	Quality-Gate debt metrics in IDE [28]	No	No	Minutes-based remediation cost, TD ratio [29] [30] [31]	Use SonarQube TD plus BI tooling
Green impact	CO$_2$ scoring & quick wins [13]	No	No	No	No	Custom extension needed
Cloud service recommender	AWS/Azure/GCP mapping [3] [5]	Limited	No	No	No	Build rule-engine atop SAST data
CWE-based future vuln prediction	Yes [9]	Research stage	No	No	No	Possible via Black Duck data science
Portfolio dashboards	Yes, out-of-box [1]	Aggregated risk dashboard [23]	Multi-app views [22]	HTML or SIEM export	Portfolios in Enterprise Edition [32]	Need external BI layer

Every value claims the referenced citation IDs directly after the text.

Building a CAST-Like Solution with Checkmarx, Black Duck, TruffleHog & SonarQube

Architectural Overview

1. **Source-Code Repos ➔**
   a) Checkmarx SAST CLI (custom code flaws)
   b) Black Duck Detect (OSS SBOM + license risk)
   c) SonarQube Runner (code quality + TD)
   d) TruffleHog pre-commit / CI hook (secrets)
2. **CI/CD Pipeline** (Jenkins, GitLab CI, or GitHub Actions) orchestrates parallel scans and pushes JSON results to an **Aggregation Layer** (ElasticSearch + Logstash or GraphQL API).
3. **Analytics & Dashboards**: Kibana, Grafana, or Power BI consolidate:
   - Cloud-blocker heuristics (custom rules on SAST + grep results)
   - OSS risks (Black Duck API)
   - Technical Debt ($) from SonarQube Web API
   - Secret counts from TruffleHog JSON
4. **Recommendation Engine**: Python micro-service maps patterns to 5 R migration categories, replicating CAST's Cloud Ready wave planner. Input variables: `cloud_blockers`, `effort_minutes`, `business_criticality`, `td_ratio`.
5. **Executive Portal**: Single-page web app (React) surfaces drill-downs; uses RBAC; export to PDF.

Step-by-Step Implementation

1. Static Code & Security Analysis

- **Checkmarx**:
  - Enable SAST + SCA modules; import preset “OWASP Top 10” rules; schedule incremental scans on every merge to `develop`[26] [15].
  - Enable *Best-Fix Location* to correlate duplicates and reduce noise (90% reduction in triage effort)[23].
- **SonarQube**:
  - Install Community or Dev Edition, activate branch analysis and pull-request decoration[33] [32].
  - Calibrate `sonar.technicalDebt.developmentCost` to align with CAST’s $3.61/LOC by setting 0.06 days per LOC (~28 minutes)[31].

2. Open-Source Risk & SBOM

- **Black Duck Detect**:
  - Scan via Build-tool integration (Maven, Gradle, npm).
  - Consume BDSA feed for ahead-of-NVD vulnerability data, mirroring CAST’s future-weakness analysis[22] [34].
  - Export CycloneDX SBOM JSON for auditing.

3. Secrets Management

- **TruffleHog**:
  - Add `trufflehog git --since-commit main --results verified --fail` as a Git pre-receive hook; CI build aborts with exit 183 if secrets found[24] [25].
  - Feed verified secret counts into the central ElasticSearch index for KPI tracking.

4. Cloud-Readiness Model

- **Rule Mining**: Create regex & AST rules in Checkmarx Custom Queries to flag:
  - File-system writes (`java.io.FileWriter`)
  - Hard-coded `java.sql.DriverManager.getConnection`
  - COM Interop for .NET.
- **Effort Estimation**: Assign T-shirt size → developer-days matrix (S = 0.5 day, M = 1 day, L = 3 days) comparable to Highlight presets.
- **Wave Algorithm (Python pseudo-code)**

```
for app in portfolio:
    score = (100 - blockers_weight) + boosters_weight
    effort = sum(b.remediation_days for b in blockers)
    if score >= 80 and effort < 10: category = "Rehost"
    elif score >= 60: category = "Refactor"
    elif score >= 40: category = "Rearchitect"
    else: category = "Rebuild"
```

5. Technical Debt Monetization

- Pull SonarQube metric `sqale_index` (minutes) via REST; multiply by blended labor rate ($800/day) to express in USD, matching CAST's executive dashboards.

6. Green Impact (Optional)

- Write a SonarQube custom plug-in that flags high-CPU loops, repeated string concatenation, or N+1 SQL iterations; map each to energy factors from Green Software Foundation guidelines.

7. Dashboards & Reporting

- Use Grafana to overlay:
  - **Cloud Readiness vs Technical Debt** scatter for rationalization.
  - **OSS Risk Heat-Map** (Apps on x-axis, CVSS on y-axis).
  - **Secrets Exposure Trend** line from TruffleHog metrics.
- PDF export via Grafana Image Renderer schedules weekly executive packs, paralleling Highlight's built-in PDF export[1].

Governance & Quality Gates

Gate	Metric	Threshold	Tool Source
Security	New critical SAST vulns	= 0	Checkmarx[26]
Maintainability	TD ratio on new code	< 3%	SonarQube[29]
OSS Risk	Components with CVSS ≥ 8	≤ 1 per app	Black Duck[21]
Secrets	Verified secret count	= 0	TruffleHog[25]
Cloud Waves	Blockers resolved before wave start	100%	Custom engine

Limitations & Future Enhancements

- **Cloud Service Recommendations**: CAST's proprietary mapping is hard to replicate perfectly; enrich the engine with AWS *Porting Assistant* APIs or Azure *Migrate* to close the gap.
- **Green Metrics Validation**: CAST relies on empirical energy models; community open data sets are still immature—results may diverge.
- **Portfolio Survey Context**: Checkmarx/Black Duck lack built-in business questionnaires; implement PowerApps or Google Forms data capture linked via app-ID.

Conclusion

While CAST Highlight delivers an integrated, turnkey SaaS for portfolio-level modernization insight, you can emulate nearly every capability by orchestrating:

- **Checkmarx** for *SAST/SCA* and custom cloud blocker rules,
- **Black Duck** for deep OSS governance and ahead-of-NVD vulnerability feeds,
- **TruffleHog** for secrets detection and validation, and
- **SonarQube** for granular technical debt economics and code-quality trendlines.

The combined toolchain, stitched together with CI/CD automation, a unified datastore, and lightweight analytics micro-services, offers an open, extensible alternative that rivals CAST Highlight's analytical breadth—while giving engineering teams full control over data residence and scan scheduling. Continuous calibration of remediation effort, debt costing, and cloud-migration heuristics will bring the stack ever closer to CAST's maturity and will future-proof your organization's software-intelligence capabilities.

⁂

1. https://content.castsoftware.com/hubfs/aws/CAST-Highlight-One-Pager.pdf
2. https://doc.casthighlight.com/about-us/
3. https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/assess-application-readiness-for-migration-to-the-aws-cloud-by-using-cast-highlight.html
4. https://cast-software.awsworkshop.io/3_detailedresults/31_cloudready.html
5. https://cloud.google.com/blog/products/application-modernization/cast-highlight-organizes-application-migrations-to-google-cloud
6. https://learn.castsoftware.com/optimize-applications-for-cloud-faster
7. https://doc.casthighlight.com/open-source-license-risk-profiles/
8. https://www.castsoftware.com/pulse/an-open-source-risk-sherpa
9. https://www.castsoftware.com/pulse/analyze-open-source-weaknesses-before-they-become-known-vulnerabilities
10. https://doc.casthighlight.com/feature-focus-portfolio-advisor-technical-debt/
11. https://www.castsoftware.com/pulse/treat-your-tech-debt-like-financial-debt-to-have-the-best-impact
12. https://www.castsoftware.com/glossary/technical-dept-estimation
13. https://learn.castsoftware.com/hubfs/pdf-files/CAST-Highlight_Overview.pdf
14. https://doc.casthighlight.com/Getting-Started-Guide.pdf
15. https://www.qiminfo.ch/en/all-about-checkmarx-the-application-security-solution/
16. https://www.blackduck.com/software-composition-analysis-tools.html
17. https://www.jit.io/resources/appsec-tools/trufflehog-a-deep-dive-on-secret-management-and-how-to-fix-exposed-secrets
18. https://en.wikipedia.org/wiki/SonarQube
19. https://checkmarx.com/blog/the-roi-of-sca-reducing-technical-debt-and-enhancing-security/
20. https://checkmarx.com/learn/sca/sca-sast-dast/
21. https://www.blackduck.com/software-composition-analysis-tools/open-source-scanning.html

22. https://www.blackduck.com/software-composition-analysis-tools/black-duck-sca.html
23. https://checkmarx.com
24. https://github.com/trufflesecurity/trufflehog
25. https://www.helpnetsecurity.com/2024/02/21/trufflehog-open-source-solution-for-scanning-secrets/
26. https://www.devopsschool.com/blog/what-is-checkmarx-and-use-cases-of-checkmarx/
27. https://www.sonarsource.com/products/sonarqube/
28. https://checkmarx.com/in-the-news/understanding-technical-debt-for-software-teams/
29. https://stackoverflow.com/questions/49179522/how-does-sonarqube-calculate-technical-debt
30. https://community.sonarsource.com/t/how-does-technical-debt-being-calculated/38455
31. https://docs.sonarsource.com/sonarqube-server/10.6/user-guide/code-metrics/modifying-technical-debt-parameters/
32. https://www.almtoolbox.com/sonarqube
33. https://k21academy.com/devops-job-bootcamp/sonarqube/
34. https://www.gartner.com/reviews/market/application-security-testing/vendor/black-duck/product/black-duck-software-composition-analysis