APPLAND

Developers are wasting valuable time on bad code.

Up to half of all developer effort[1] is spent trying to understand and remediate code quality issues that impact performance, reliability, maintainability and security. High-performing software organizations may have code quality gates in place, however these don't cover *structural code quality issues* -- which are just as harmful and more complicated to identify than static code issues.

Automated analysis of runtime code behavior pre-deployment can help.

Use AppMap to automatically scan for structural code issues in Dev and CI/CD that static code analyzers miss. Identify performance flaws with dynamic architectural analysis and pre-production performance data -- no production load required. Developers can see the root cause of issues via code-linked maps and find security vulnerabilities before they ship, all from within their existing dev environment.

Scan for structural code issues like these with AppMap:

Performance

- Filtering and sorting data in memory (rather than in the database).
- Repeated SQL queries due to absence of eager loading.
- Repeated API requests due to absence of batch retrieval.
- Repeated retrieval of the same data from API, due to failure to use cache headers.
- Repeated execution of a long-running query within the same request or session.
- Request time exceeds a threshold due to too much data processing inline with the request. Time-intensive operations should be moved to a background job.
- Number of SQL joins exceeds a threshold (CWE-1049).
- Query without necessary filter or limit.
- Unpooled connection (CWE-1072).

Reliability

- Web service request returns a 500 error code (CWE-394).
- Web client request has an excessively long timeout.
- Lack of debug-level logging (CWE-778).
- No distinction between auditing and logging.

Maintainability

- DAO object is saved without being validated (CWE-20).
- Web controller performs direct database access.
- Violation of package dependency constraint (modularity breakdown).
- Code path is utilized by a single customer only.
- Introduction of a circular dependency (CWE-1047).
- Duplication of data across database fields (related to CWE-1041).
- GET request creates or updates data (other than logging or audit) (related to CWE-650).

Security

- Non-public web request returns 2xx code without applying authentication (CWE-306).
- Authorization precedes authentication (CWE-551).
- Framework-provided sanitizer is not applied to a request parameter.
- De-serialization of non-sanitized input (e.g. from a cache).

[1] https://stripe.com/reports/developer-coefficient-2018

APPLAND

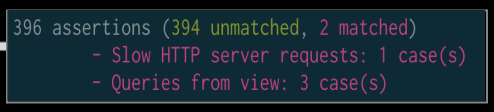
Open up new dimensions of code structure to automated analysis.

AppMap represents runtime code execution as a canonical data structure, which enables the use of generic match patterns to identify problems. Labels add additional meaning to the data and increase the power of AppMap matching patterns. AppMap integrates deeply with web frameworks to automatically label code.

	Static Code Analysis	AppMap
Scope of problems detected	Single function, or local function environment	Entire application or service
Problem types	Modularity, style	Performance, Reliability, Maintainability + Security
Environments	Dev + CI/CD	Dev, CI/CD + Staging
Microservice Analysis	No	Yes
Root cause	Yes	Yes
False Positive Scans	High	Low
Runtime Execution Analysis	No	Yes
Time-to-scan large codebases	Hours to days additional to CI job	Fixed % overhead to CI job

AppMap workflow

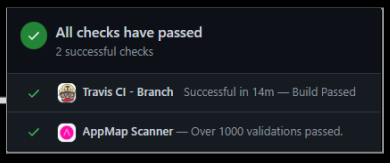
1



Analyze every code change

A dev implements a new feature or fixes a bug. Continuous Integration checks the new commit by validating performance, reliability and maintainability metrics.

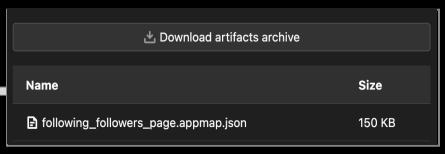
2



Report check status to GitHub/GitLab

AppMap Scanner check status is reported directly to a pull request, so the dev can see if there are any violations caused by their change.

3



See AppMaps as job artifacts

All AppMaps used to run checks are stored as CI job artifacts and could be downloaded or opened directly in the browser to review the issues.

4

Fix the findings

A dev can quickly switch to AppMap Trace View to review the event that triggered a violation and fix it.

Talk to us about the structural quality of your code!

Follow us on Twitter at @landofapps.