AUBURN

Static Application Security Testing (SAST)

SAST Overview

AUBURN

- Developers typically outnumber security staff!
- White Box testing approach - the internals of an application are available to be tested.
- Scans happened before the source code is compiled - does not require a working application
- Ability to identify high percentage of known vulnerabilities (CVE, CWE)
- Issues can be tracked and manage in an organized way providing metrics and dashboards
- Scan millions of lines of code in a matter of minutes (only looks at code changes and the impact on existing code)
-

Static Scanning Capabilities

AUBURN

- Quality
  - Catches typical coding issues that might causes application errors
- Security
  - Inspects for known coding or configuration vulnerabilities that have the potential to introduce vulnerabilities
- Good Coding Practice
- Adherence to Standards
  - Custom rules can be defined to catch you own defined coding standards and checks
- Education
  - Tools typically provide a good description on why a condition has been flagged and a recommend solutions.
  - For less experienced software engineers this can be a good method of learning

Static Scanning Advantages

AUBURN

- Can be applied early in the development cycle
- Access to entire source code
  - Automatic detection of source types
  - Applies rules based on source code type.
- Cover entire code base in a small amount of time
  - Ideal for automated build pipelines
- Identify exact location in code of isse
- Quick turn around for fixes
  - Fixes are usually straightforward and small. Tools suggest what code should look like
  - Integration into IDEs and Version Control Tools
- Good documentation pointing CWE or CVE references.
- Studies have shown SAST scanning tools have a high probability of find client side (browser) and server side vulnerabilities

Static Scanning Disadvantages

AUBURN

- False positives/negatives
  - Typically rules need to be tweaked to apply to your coding standards
  - Initial scans on existing large code bases can result in 100's or 1000's of issues
- Weak on operation/runtime issues
- If scans are manually triggered there can be a delay and issues build up because it is not automated triggered on code changes/commits
- False sense of security once automated
  - One must understand the limitations of SAST scans and not believe it catches all code level issues.
- Only as good as configured rules
  - Rules should be updated on an ongoing bases to encompass new known vulnerabilities

SAST vs DAST Security Testing

AUBURN

SAST

- White Box Testing
- Requires Source Code
- Early SLDC Detection
- Less Expensive to Fix
- No Runtime Issue Discovery
- Supports Many Languages

DAST

- Black/Grey Testing
- Requires Running Application
- Late SLDC Detection
- More Expensive to Fix
- Discover Run Time and Environmental Issues
- Limited application types (HTTP, UI)

Common Vulnerabilities Found

AUBURN

- Hard coded sensitive information
- Command injection
- SQL injection
- Cross site scripting
- Buffer overflow (lower level languages)
- Poor coding practices
- Exception handling

SAST Types & Severity (SonarQube)

AUBURN

Issues Types

Bug – A coding mistake that can lead to an error or unexpected behavior at runtime.

Vulnerability – A point in your code that's has the potential to cause a security related issues.

Code Smell – A maintainability issue that may cause a deeper issue. Typically easy to spot such as long methods, deprecated methods used, etc.

Severity

Blocker - Bug with a high probability to impact the behavior of the application in production: memory leak, unclosed JDBC connection.

Critical - Either a bug with a low probability to impact the behavior of the application in production or an issue which represents a security flaw: empty catch block, SQL injection.

Major - Quality flaw which can highly impact the developer productivity: uncovered piece of code, duplicated blocks, unused parameters, etc.

Minor - Quality flaw which can slightly impact the developer productivity: lines should not be too long, "switch" statements should have at least 3 cases, etc.

Info - Neither a bug nor a quality flaw, just a finding.

SAST Scanning Metrics

AUBURN

- **Complexity** - Number of paths through the code, hard to understand code
- **Duplication** - Number of duplicated blocks of lines.
- **Issues** - New, Open, False Positives, Confirmed, Deferred, Marked Ok.
- **Maintainability** - “Code smells”, estimated time to remediate
- **Reliability -** Bugs, new bugs, rating based on number & severity
- **Security -** Vulnerabilities, number and severity, overall rating, number reviewed and closed
- **Size** - LOC, LOC per language, statements classes, comments, directories, file types.
- **Test Coverage** - Conditions, number of unit test, test pass, test duration, test coverage

Quality Gates

AUBURN

- Quality gates based on metric threshold can prevent software from being built unless a minimum threshold is met.
- For security type issues:
  - Hotspots - identified security issues needing review
  - Rating - overall rating based on type and severity of issues.

Conditions on New Code

Metric	Operator	Value
Coverage	is less than	80.0%
Duplicated Lines (%)	is greater than	3.0%
Maintainability Rating	is worse than	A
Reliability Rating	is worse than	A
Security Hotspots Reviewed	is less than	100%
Security Rating	is worse than	A

Typical SAST Process

AUBURN