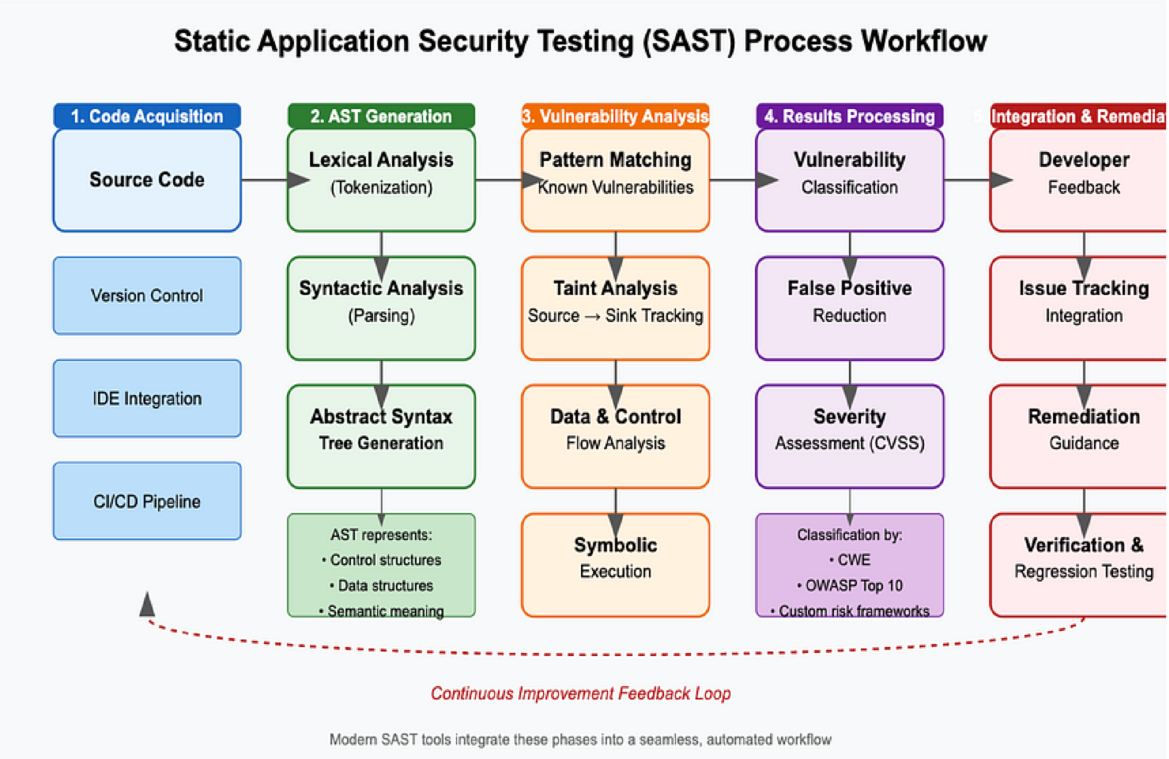
Security Checklist for Software Developers – Lab. Exercise

Several studies and industry guidelines underscore the importance of selecting SAST tools that align with both the development stack and the organization's security objectives.



Static Application Security Testing (SAST) Process Workflow

Modern SAST tools integrate these phases into a seamless, automated workflow

As reported in OWASP guidelines, tool effectiveness is highly context-dependent: while some tools excel at finding low-level memory issues, others are more suitable for detecting high-level code quality issues or logic vulnerabilities. For this reason, a comparative evaluation was conducted to identify the most appropriate tools to integrate into the prototype framework.

Tool Selection Criteria:

- Language coverage (Python, JS, Java, Go, C/C++, PHP, Ruby, Typescript)
- Open-source, lightweight, CLI-ready
- Good documentation, active community
- Suitable for automation and integration

Related projects:

- **CodeQL**: semantic code querying from GitHub.
- **Semgrep**: rule-based, highly customizable, broad language support.
- **SonarQube**: enterprise-grade with dashboards & CI/CD integration.

- **Checkmarx, Veracode, Synk:** commercial static analyzers for enterprise use

A first set of well known solutions:

FLAWFINDER

FlawFinder is a quick yet simple and efficient tool that scans C/C++ source code. The scan analyzes calls to typical vulnerable library functions. It is run from the command line. Its output can be easily customized.

Typical error types detected:

• Library function calls that create buffer overflow vulnerabilities (gets, strcpy, sprintf, ...)

• Calls to library functions potentially vulnerable to string formatting attacks (sprintf, printf, ...)

• Potential problems resulting from mishandling of files.

https://github.com/david-a-wheeler/flawfinder

https://www.geeksforgeeks.org/how-to-use-flawfinder-python-tool-to-find-vulnerabilities-in-c-cpp-code/

Visual Code Grepper (VCG)

Visual Code Grepper is an automatic code security review tool that handles multiple programming languages such as C/C++, Java, C#, VB, and PL/SQL. The tool offers some features that make it useful for anyone who conducts code security reviews, such as:

- • Performing some complex checks
- • A configuration file for each language which basically allows you to add any incorrect functions, commands or other text/scripts you want to search for.
- • The tool attempts to find a range of approximately 20 expressions typically used by developers within comments and which may indicate broken or simply unfinalized code (typical expressions such as "ToDo", "FixMe", "Kludge ", etc.)
- • Provides a pie chart (for the entire code base and for individual files) showing the relative proportions of code, whitespace, comments, ToDo-style comments, and bad code.
- • VCG includes techniques to identify buffer overflow risks and operations that present data format risks (e.g. signed/unsigned).

https://github.com/nccgroup/VCG

PYLINT

Pylint is a tool that checks Python code for errors, tries to enforce a coding standard, and looks for code that could pose security risks. Similar to pychecker, except that pychecker doesn't explicitly care about coding style. Pylint provides a series of messages as it analyzes the code, as well as some statistics on the number of warnings and errors found in different files. Messages are classified into various categories.

https://github.com/PyCQA/pylint

Microsoft CAT.NET

Microsoft CAT.NET is an older but interesting static analysis tool for analyzing software security issues. The solution uses an algorithm based on results and experience from Work Teams and Microsoft Research. It is free to download and is actively developed by the Information Security Tools team. The tool is relatively old but the link provides interesting useful insights:

https://www.codeguru.com/csharp/securing-your-asp-net-application-using-microsoft-cat-net/

A group of further solutions:

Tool	Language(s)	Category
Bandit	Python	Security
Flake8	Python	Style
Pyright	Python	Type safety
Pylint	Python	Style
Checkstyle	Java	Style
Spotbugs	Java	Security/Bug Detection
PHPStan	PHP	Type safety
Bearer CLI	Multi	Multi-language
Gosec	Go	Security
Staticcheck	Go	Bug-Detection/Style
CppCheck	C/C++	Bug-Detection/Style
FlawFinder	C/C++	Security
ESLint	JavaScript	Style
Njsscan	JavaScript	Security
Rubocop	Ruby	Style
Brakeman	Ruby (Rails)	Security

Assignments:

This exercise aims at practicising techniques and solutions to support the analysis of software code with the objective to identify any security problems.

1. Please download one or more of the tools listed and test their operation and functionality.
2. Provide a comment regarding the tools analyzed, in particular: ease of use, outputs, configuration possibilities, time required for the analysis, ...
3. Look for other useful tools and propose a comparative analysis.

Additional Interesting Links:

https://marketplace.eclipse.org/content/cigital-secureassist

https://www.synopsys.com/software-integrity/security-testing/software-composition-analysis.html

https://www.aquasec.com/products/trivy/