Fundamentals of Application Security Testing Tools

Dr. Thomas P. Scanlon

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213
We Have Lawyers

Copyright 2021 Carnegie Mellon University.

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8702-15-D-0002 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

The view, opinions, and/or findings contained in this material are those of the author(s) and should not be construed as an official Government position, policy, or decision, unless designated by other documentation.

References herein to any specific commercial product, process, or service by trade name, trade mark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by Carnegie Mellon University or its Software Engineering Institute.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN "AS-IS" BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.


DM21-0408
AST Tools Benefits

- Increase speed, efficiency, and coverage paths
- Repeatable
- Scale well
- Find known vulnerabilities, issues and weaknesses
- Triage and classify findings
- Assist in remediation workflow, especially verification
- Can be used for correlation and identification of trends and patterns
Types of Application Security Testing Tools

https://insights.sei.cmu.edu/blog/10-types-of-application-security-testing-tools-when-and-how-to-use-them/
Application Security Testing Tools Pyramid

- Application Security Testing Orchestration (ASTO)

  - Correlation Tools
  - Test Coverage Analyzers

  - Mobile Application Security Testing (MAST)
  - Interactive Application Security Testing (IAST) & Hybrid Tools
  - Application Security Testing as a Service (ASTaaS)

- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST)
- Origin Analysis / Software Composition Analysis (SCA)
- Database Security Scanning
Application Security Testing Tools Pyramid

- Application Security Testing Orchestration (ASTO)
- Correlation Tools
- Test Coverage Analyzers
- Mobile Application Security Testing (MAST)
- Interactive Application Security Testing (IAST) & Hybrid Tools
- Application Security Testing as a Service (ASTaaS)
- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST)
- Origin Analysis / Software Composition Analysis (SCA)
- Database Security Scanning
Static Application Security Testing (SAST)

- “Examine source code (at rest) to detect and report weaknesses that can lead to security vulnerabilities” – NIST¹

- Source code analyzers can run on non-compiled code checking for things like numeric errors, input validation, race conditions, path traversals, pointers and references, and more

- Binary and Byte Code analyzers do the same on built and compiled code

- Some tools run on source only, compiled only, or both


Can be thought of as white-hat or white-box testing
CWEs and CVEs

Common Weakness Enumerations (CWEs)

CWEs are classes of problems

Does not focus on a specific product, system, or piece of software

CWEs represent concepts/behaviors software developers should avoid

Examples: Integer Overflow; Use of Hard-coded credentials; Generation of Predictable Numbers

Common Vulnerabilities and Exposures (CVEs)

CVEs are instances of problems (instances of CWEs in some respects)

Does pertain to a specific product, system, or piece of software

CVEs represent weaknesses software developers could bring into their software via use of & inclusion of tools, libraries, products, other software

Example: CVE-2015-3429 Cross-site scripting (XSS) vulnerability in example.html in Genericons on Debian Linux before 3.3.1, as used in WordPress before 4.2.2, allows remote attackers to inject arbitrary web script or HTML via a fragment identifier.
Application Security Testing Tools Pyramid

- Application Security Testing Orchestration (ASTO)
- Correlation Tools
- Test Coverage Analyzers
- Mobile Application Security Testing (MAST)
- Interactive Application Security Testing (IAST) & Hybrid Tools
- Application Security Testing as a Service (ASTaaS)
- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST)
- Origin Analysis / Software Composition Analysis (SCA)
- Database Security Scanning
Dynamic Application Security Testing (DAST)

• “Detect conditions indicative of a security vulnerability in an application in its running state” – Gartner¹

• DAST tools run on operating code detecting issues with interfaces, requests, responses, scripting (i.e. JavaScript), data injection, sessions, authentication and more

• Fuzzing is throwing known invalid and unexpected test cases at an application, often in large volume

---

Gartner¹ - http://www.gartner.com/it-glossary/dynamic-application-security-testing-dast

¹Dynamic Application Security Testing (DAST)
Can be thought of as black-hat or black-box testing
Application Security Testing Tools Pyramid

- Application Security Testing Orchestration (ASTO)
- Correlation Tools
- Test Coverage Analyzers
  - Mobile Application Security Testing (MAST)
  - Interactive Application Security Testing (IAST) & Hybrid Tools
  - Application Security Testing as a Service (ASTaaS)
- Static Application Security Testing (SAST)
- Dynamic Application Security Testing (DAST)
- Origin Analysis / Software Composition Analysis (SCA)
- Database Security Scanning
Origin Analysis / Software Composition Analysis (SCA)

- SCA tools examine software to determine the origins of all components and libraries
- Most effective finding common and popular libraries and components, particularly open source pieces
- Compares modules found in code to list of known vulnerabilities
- Almost all use NIST National Vulnerability Database CVEs: [https://nvd.nist.gov/](https://nvd.nist.gov/)
- Finds components that are out of date and/or have patches available
- Tools can run on source code, byte code, binary code or some combination

“Software Governance processes that depend on manual inspection are guaranteed to fail.”

- Diego Lo Guidice, Forrester

---

Application Security Testing Tools Reference Model
CI/CD Development Project

Correlation Tools & ASTO

- Remediate
- If Tests Fail, Report Findings and Block Release

Developer Commits Code

- SCA
- SAST
- If Tests Pass, Compile & Build

- DAST
- IAST
- If Tests Pass, Release

Database Security Scanner
A Simple AST Decision Flow

If I can only implement one AST tool, which type should I use?

1. Is the application written in-house?
   - Yes: Are a lot of 3rd party and open source components used?
     - Yes: SAST
     - No: SCA
   - No: Do you have access to the source code?
     - Yes: DAST
     - No: NO

DISTRIBUTION STATEMENT A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.
AST Tool Type Decision Factors
AST Tool Type Decision Factors Summary

- Examining each factor will allow you to build a list of AST Tool types to consider.
- Some factors may push you to a certain type and other factors will push you away from that tool type.
- Ideally you will implement a combination of tools. SAST, DAST, and SCA should be used in combination whenever possible. Use IAST and Hybrid tools if needed to get the most coverage.
- In cases where only one or two tool types can be considered, the decision factors should help you prioritize what can be done.
- A strong understanding of traditional SAST, DAST, and SCA is useful for make decisions on MAST, IAST, and ASTaaS
- Correlation, Test Coverage and ASTO tools can improve the performance and impact of the other AST tool types.