



# 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](mailto:permission@sei.cmu.edu).

Carnegie Mellon® and CERT® are registered in the U.S. Patent and Trademark Office by Carnegie Mellon University.

DM21-0408

# AST Tools Benefits



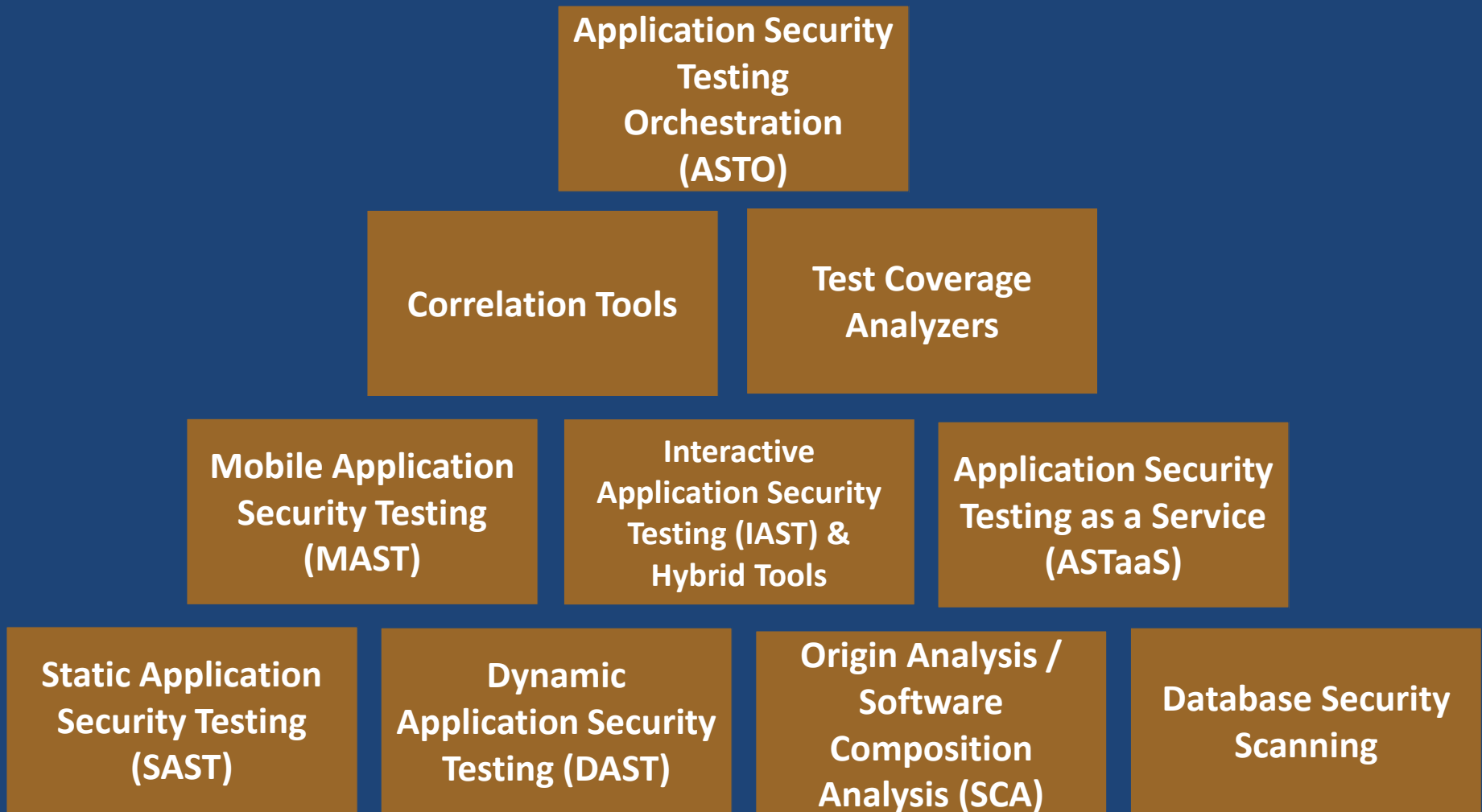
- **Increase speed, efficiency, and coverage paths**
- **Repeatable**
- **Scale well**
- **Find known vulnerabilities, issues and weaknesses**
- **Triage and classify findings**
- **Assist in remediation workflow, especially verification**
- **Can be used for correlation and identification of trends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

# Types of Application Security Testing Tools

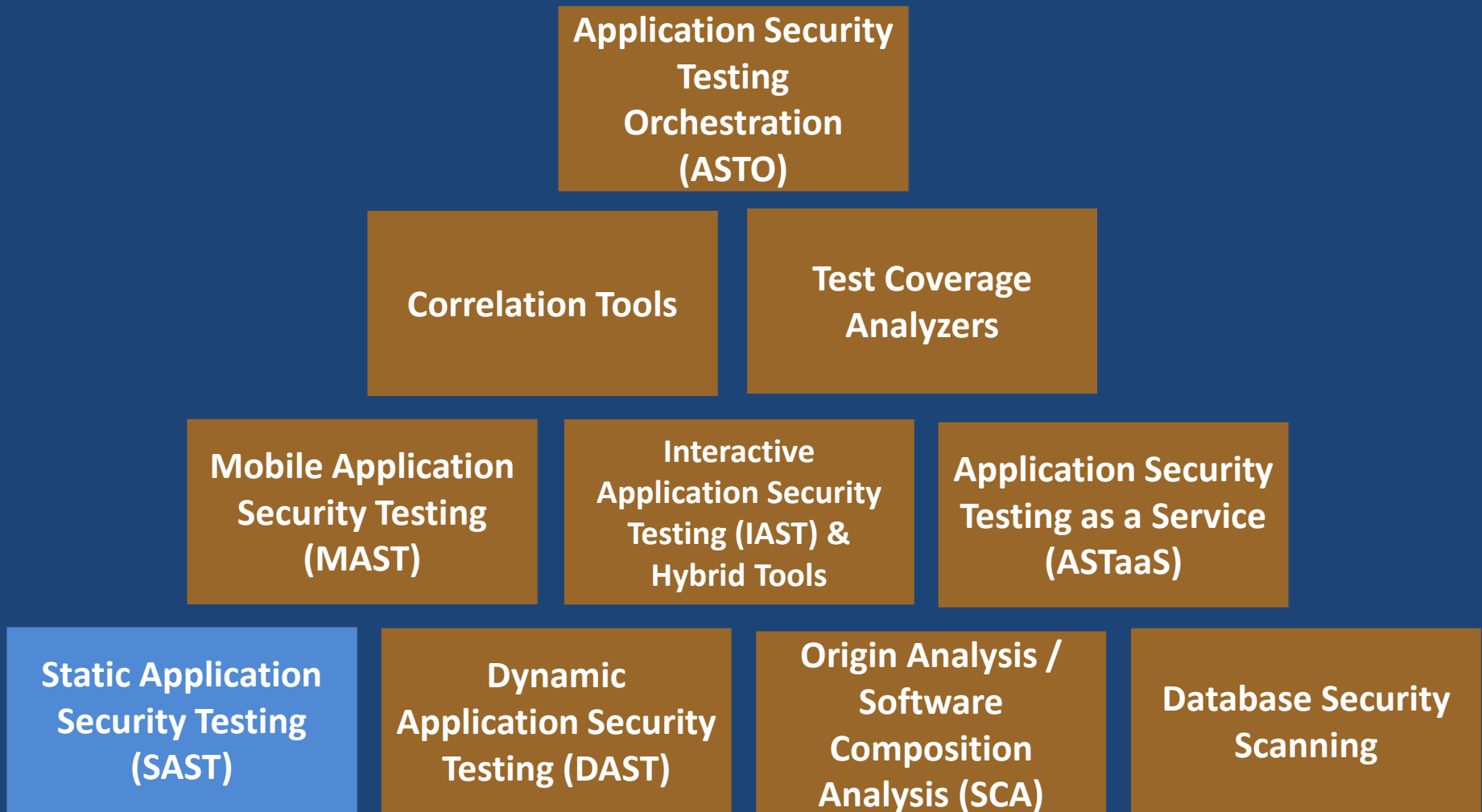
The screenshot shows the top of a web page from Carnegie Mellon University's Software Engineering Institute (SEI). The header includes the university name and a search bar. Below is a navigation menu with links for 'About', 'Our Work', 'Publications', 'News and Events', 'Education and Outreach', and 'Careers'. A large red banner features the 'SEI Blog' logo. The breadcrumb trail reads 'SEI > Publications > Blog > 10 Types of Application Security Testing Tools: When and How to Use Them'. The article title is '10 Types of Application Security Testing Tools: When and How to Use Them', written by Thomas Scanlon on July 9, 2018. A short introductory paragraph discusses the prevalence of software-related problems and the need for application security testing (AST) tools. On the right side, there are sections for 'PUBLISHED IN' (CERT/CC Vulnerabilities), 'TAGS' (cyber missions, cybersecurity, software and information assurance, testing, vulnerability analysis, security vulnerabilities), and 'SHARE' buttons.

<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 Tools Pyramid



# Static Application Security Testing (SAST)

Static Application  
Security Testing  
(SAST)

*Can be thought of  
as white-hat or  
white-box testing*

- “Examine source code (at rest) to detect and report weaknesses that can lead to security vulnerabilities” – NIST<sup>1</sup>
- 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

NIST<sup>1</sup> - [https://samate.nist.gov/index.php/Source\\_Code\\_Security\\_Analyzers.html](https://samate.nist.gov/index.php/Source_Code_Security_Analyzers.html)

# 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)

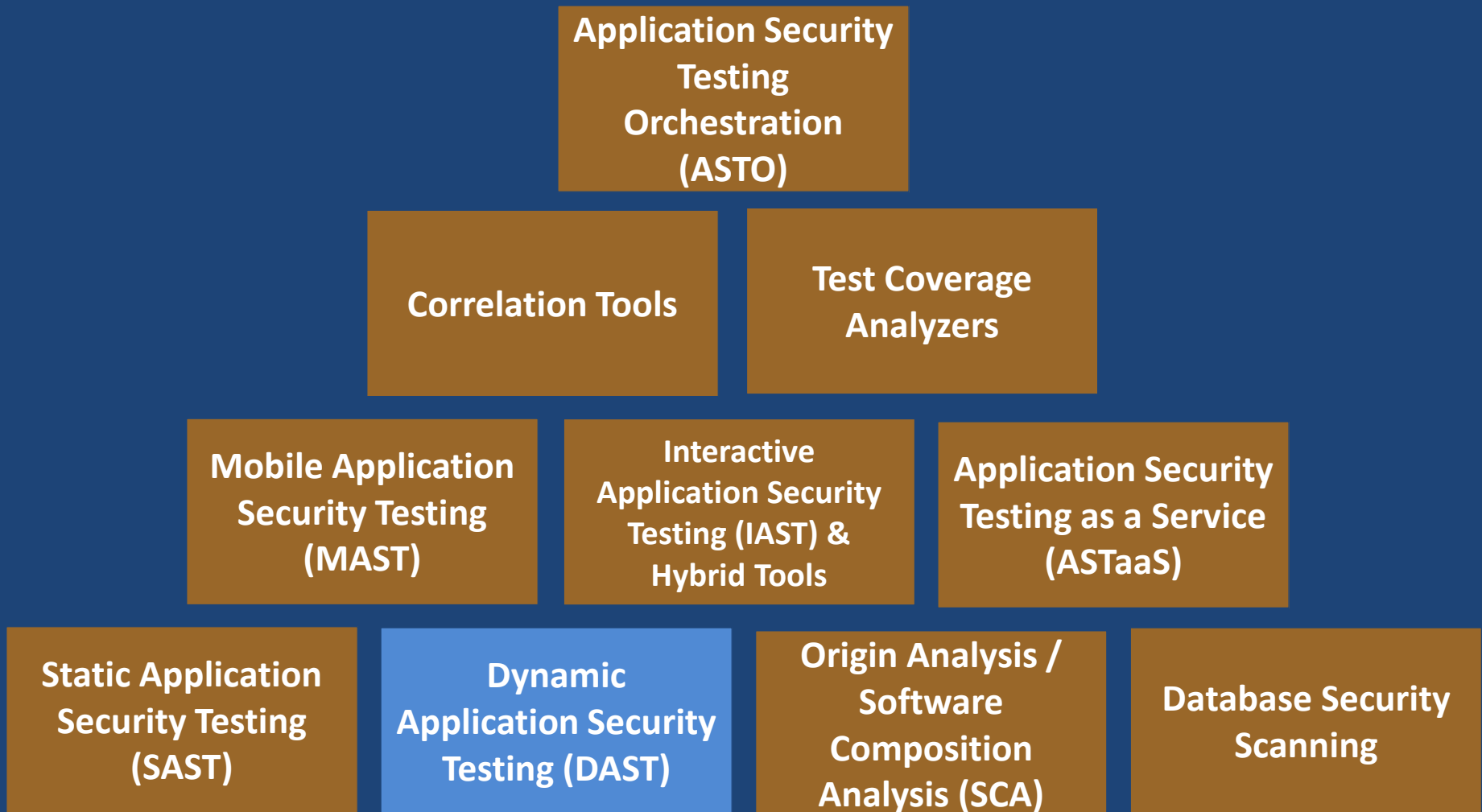
CVEs are instances of problems (instances of CWEs in some respects)

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



# Dynamic Application Security Testing (DAST)

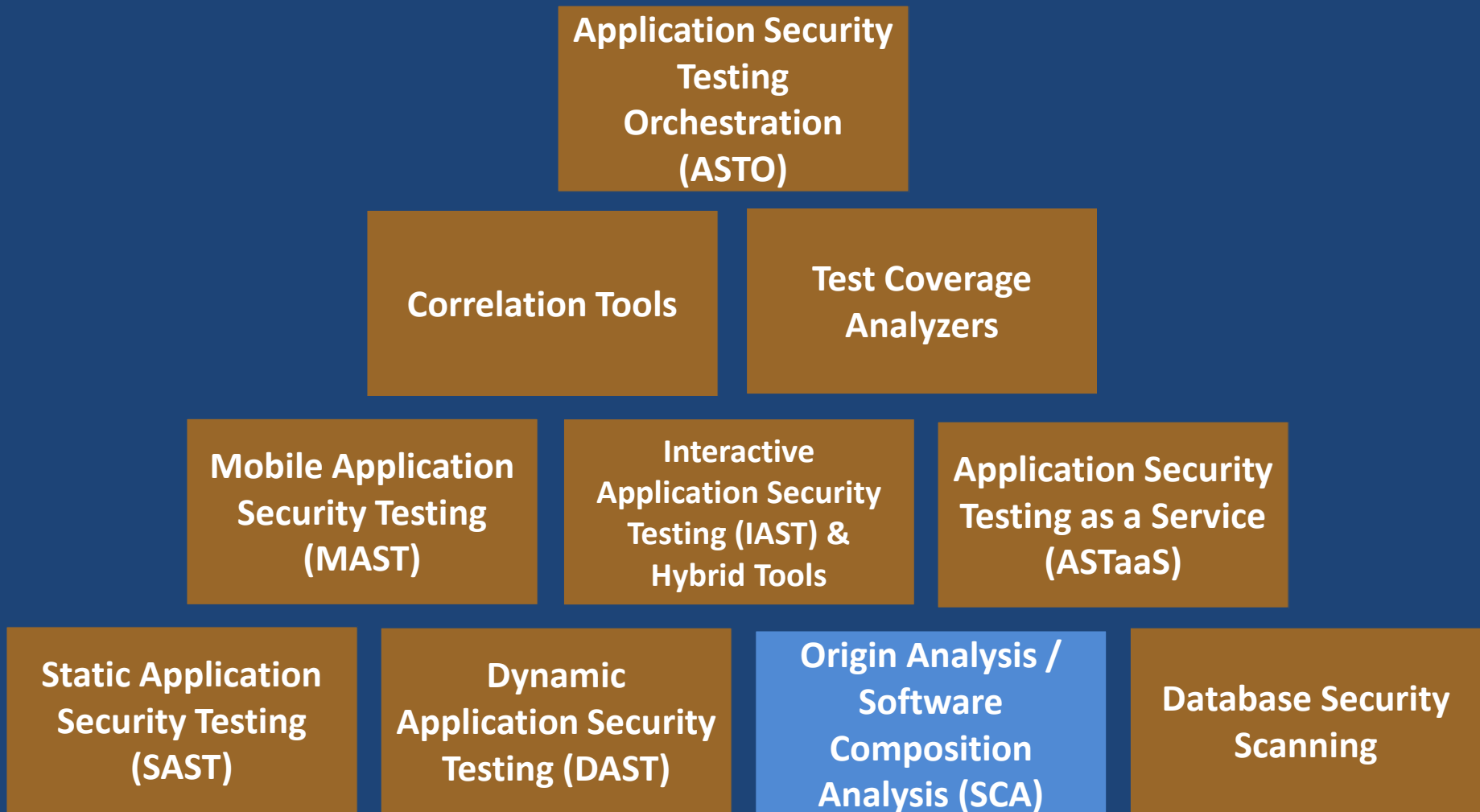
Dynamic  
Application Security  
Testing (DAST)

*Can be thought of  
as black-hat or  
black-box testing*

- “Detect conditions indicative of a security vulnerability in an application in its running state” – Gartner<sup>1</sup>
- 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

Gartner<sup>1</sup> - <http://www.gartner.com/it-glossary/dynamic-application-security-testing-dast>

# Application Security Testing Tools Pyramid



# Origin Analysis / Software Composition Analysis (SCA)

Origin Analysis /  
Software  
Composition  
Analysis (SCA)

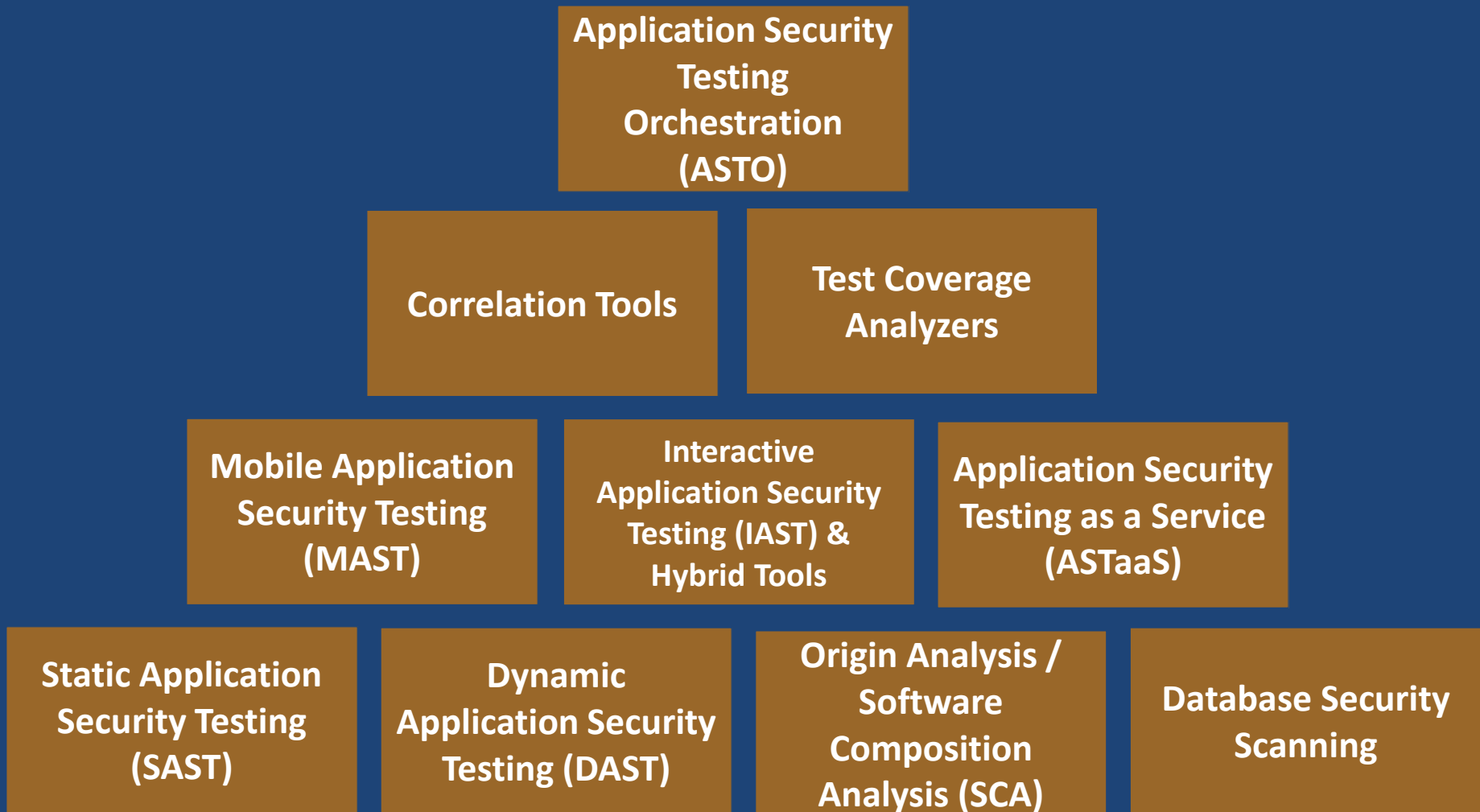
***“Software Governance processes that depend on manual inspection are guaranteed to fail.”***

***- Diego Lo Guidice, Forrester<sup>1</sup>***

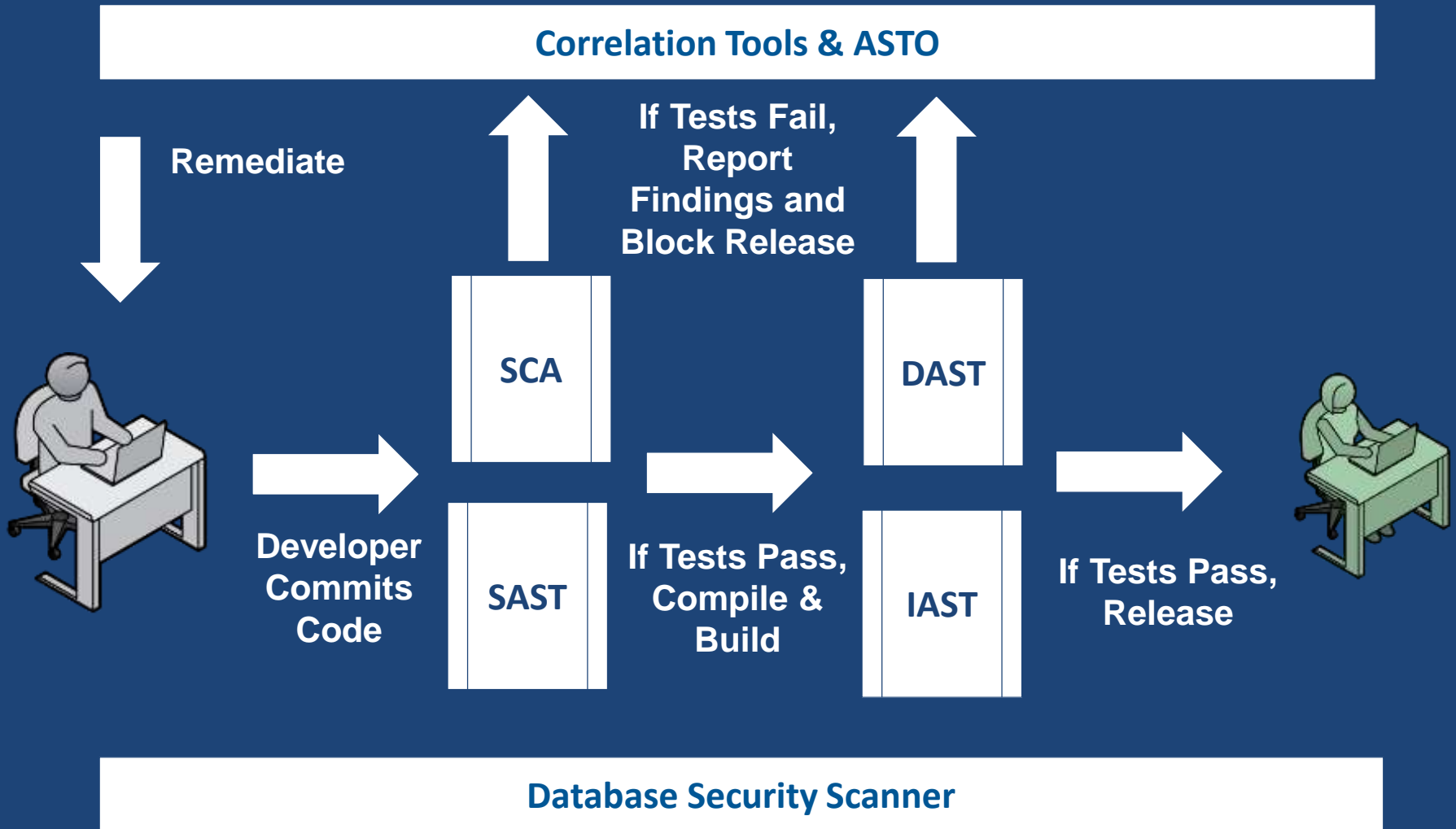
- SCA tools examine software to determine the origins of all components and libraries
- Most effective finding common and popular libraries and components, particularly open source pieces
- Compares modules found in code to list of known vulnerabilities
- Almost all use NIST National Vulnerability Database CVEs: <https://nvd.nist.gov/>
- 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

1 – Lo Guidice, Diego. Use DevOps And Supply Chain Principles To Automate Application Delivery Governance. *Forrester*. November 10, 2016.

# Application Security Testing Tools Pyramid

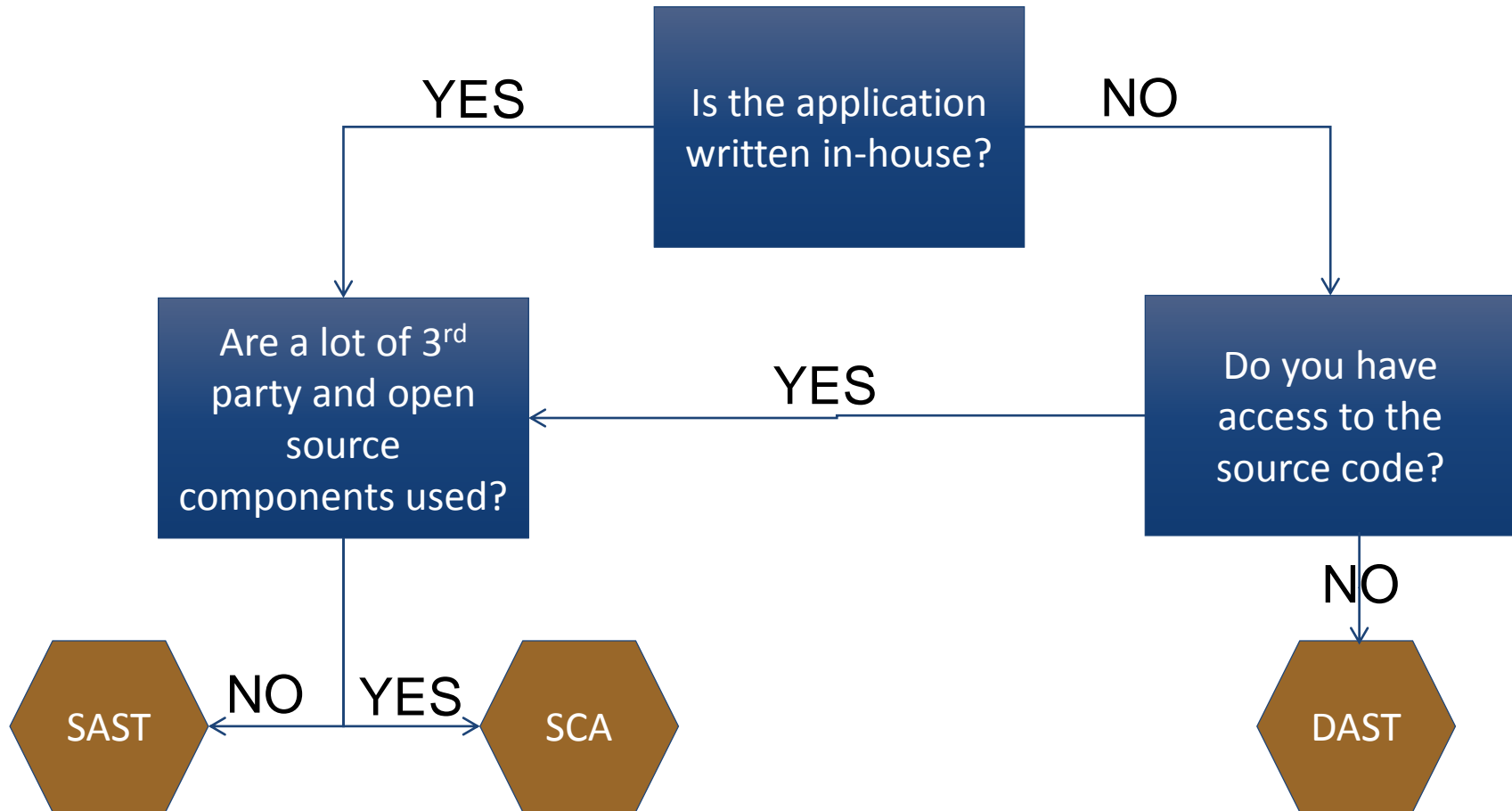


# Application Security Testing Tools Reference Model CI/CD Development Project



# A Simple AST Decision Flow

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



# 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.**