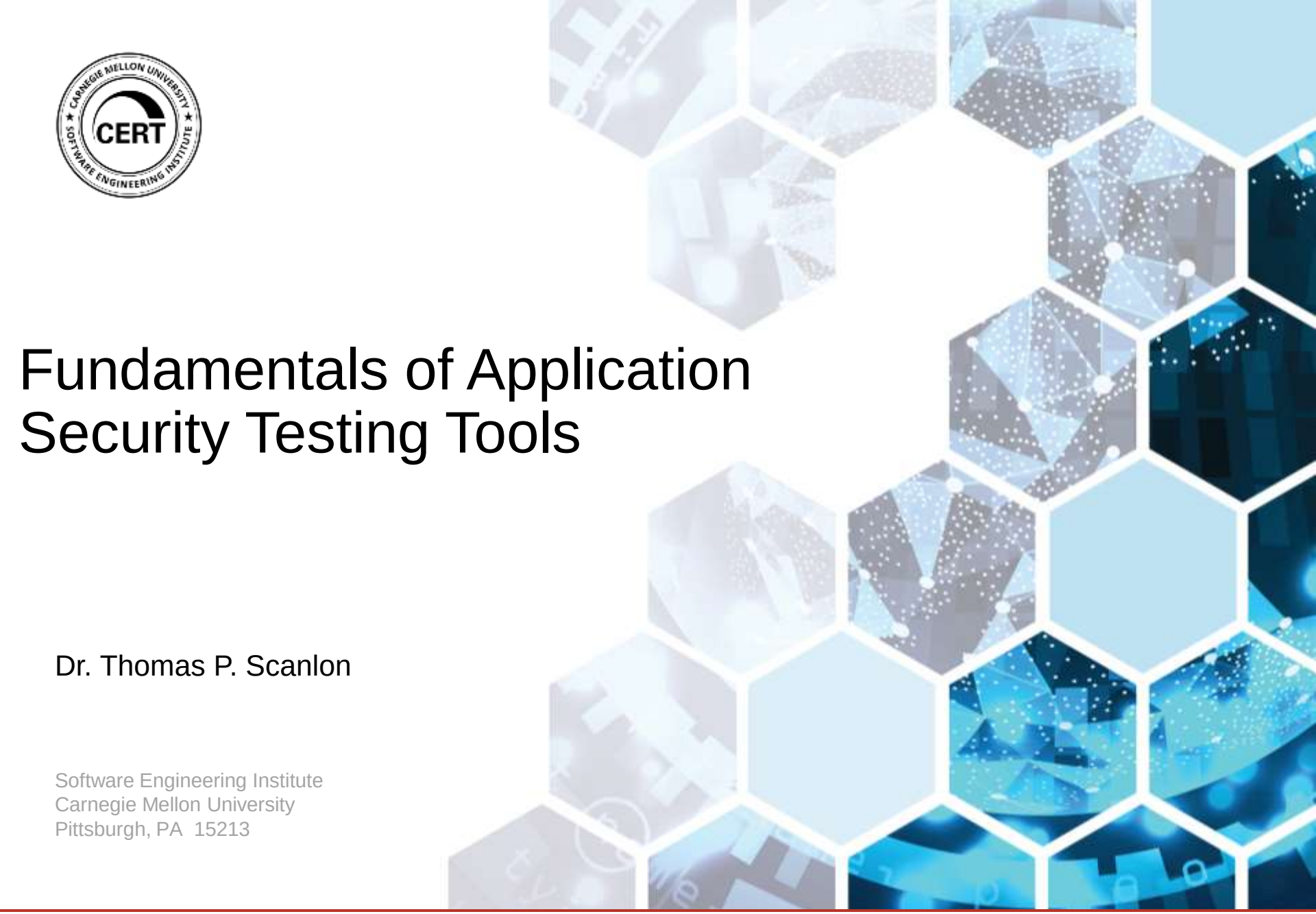


Fundamentals of Application Security Testing Tools

Dr. Thomas P. Scanlon

Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

We Have Lawyers

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.

DM21-0408

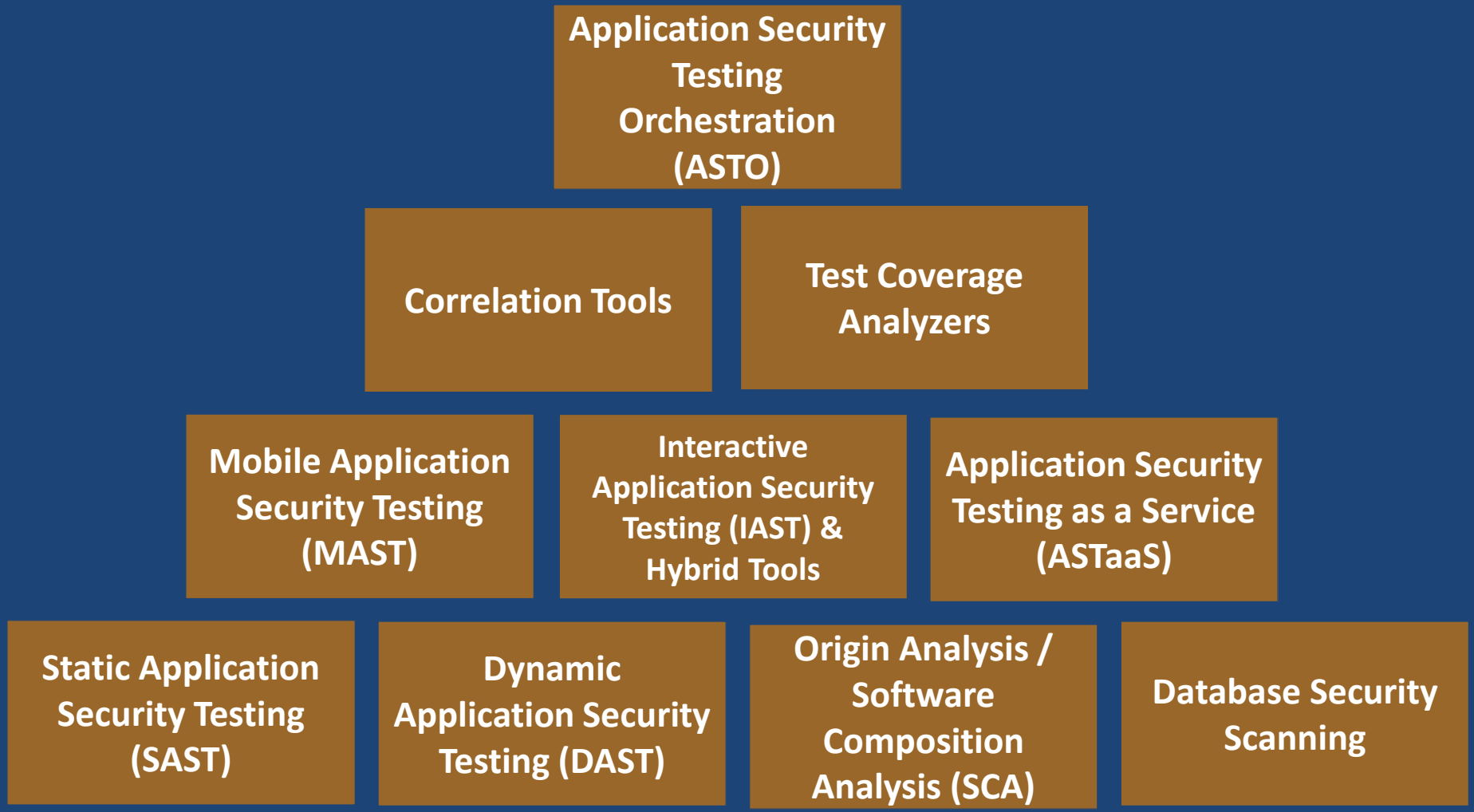
AST Tools Benefits

- **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**

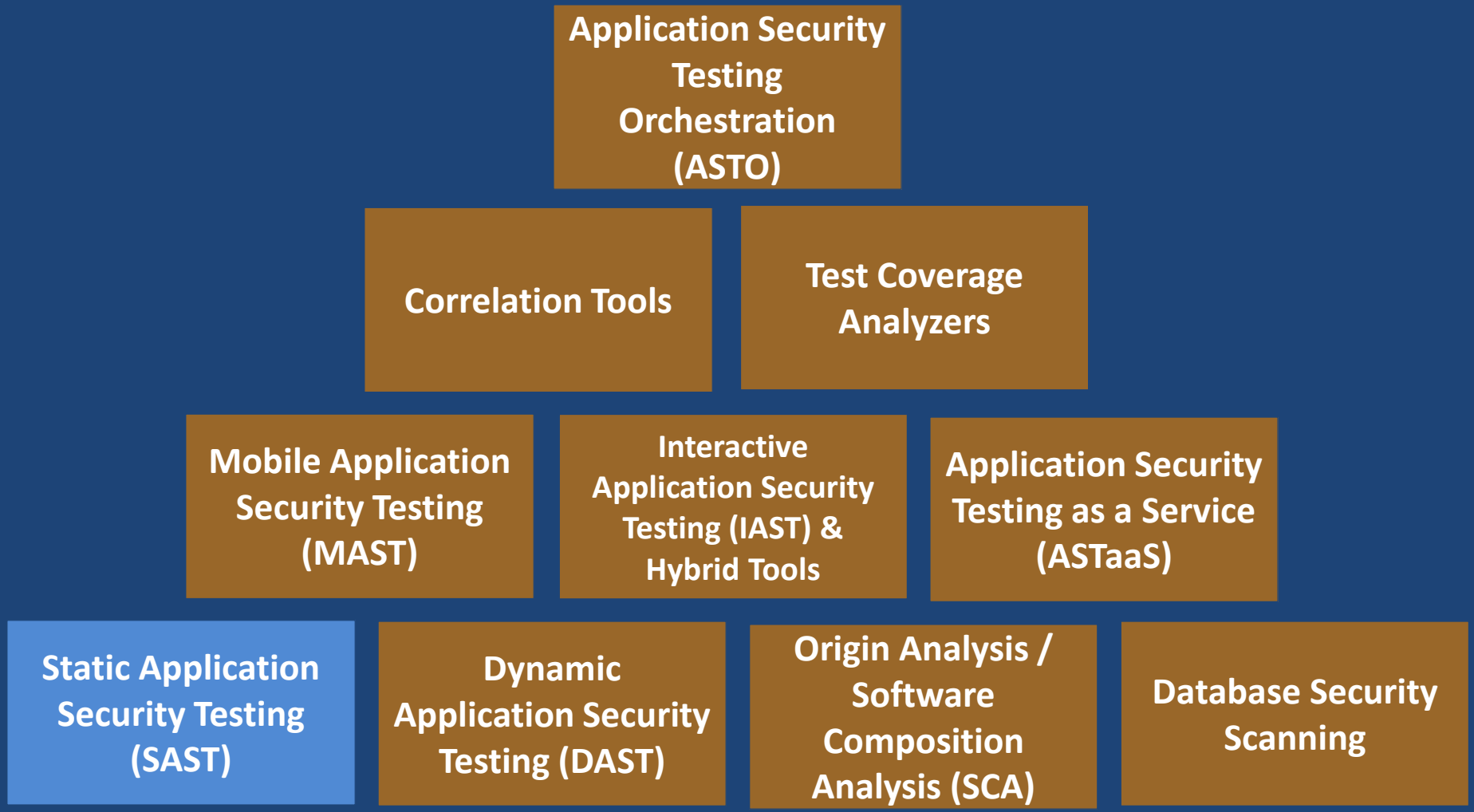
Types of Application Security Testing Tools

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 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[1]
- 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[1] - 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)

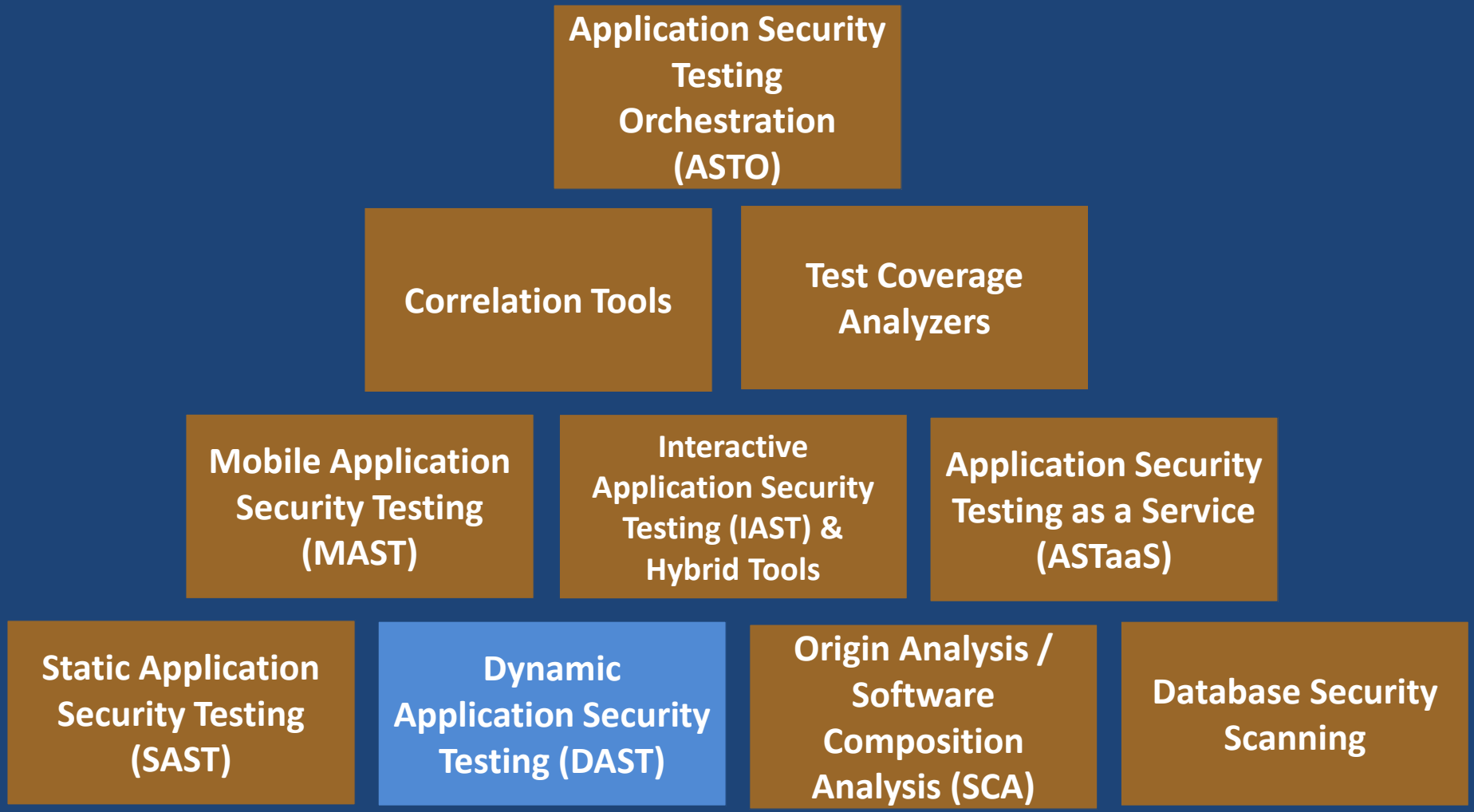
CVEs are instances of problems (instances of CWEs in some respects)

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)

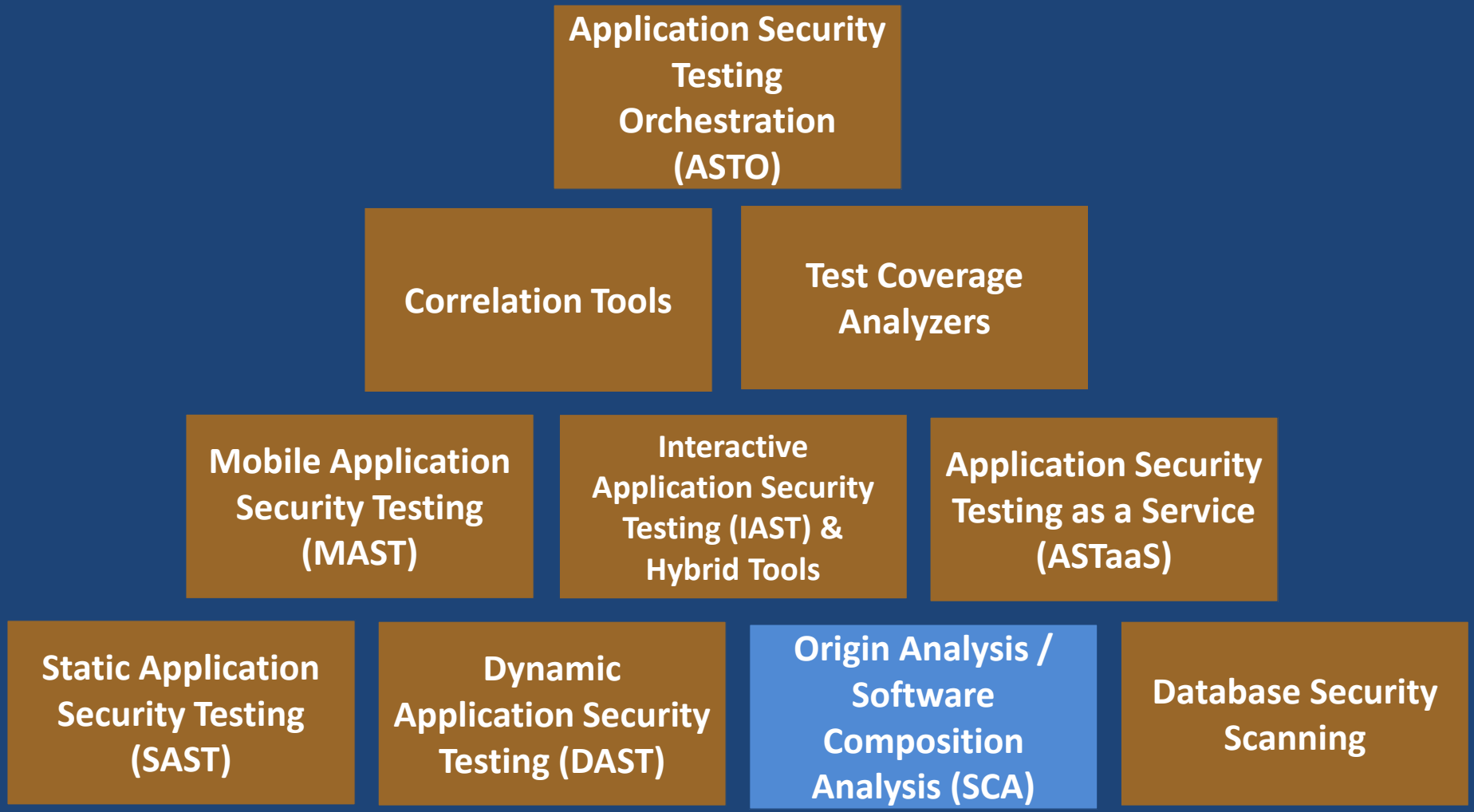
Dynamic Application Security Testing (DAST)

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[1]
- 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[1] - 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)

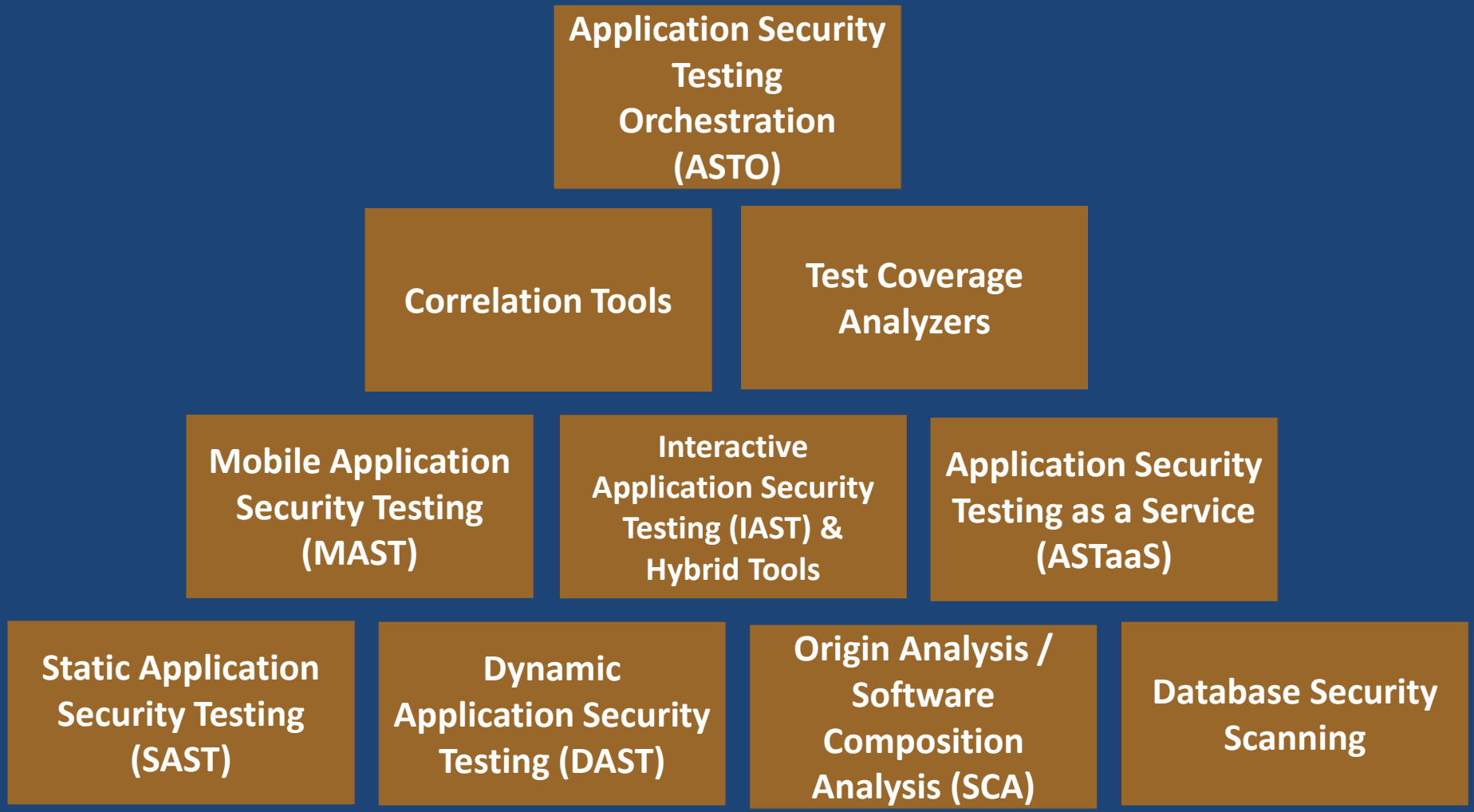
"Software Governance processes that depend on manual inspection are guaranteed to fail."

- Diego Lo Guidice, Forrester[1]

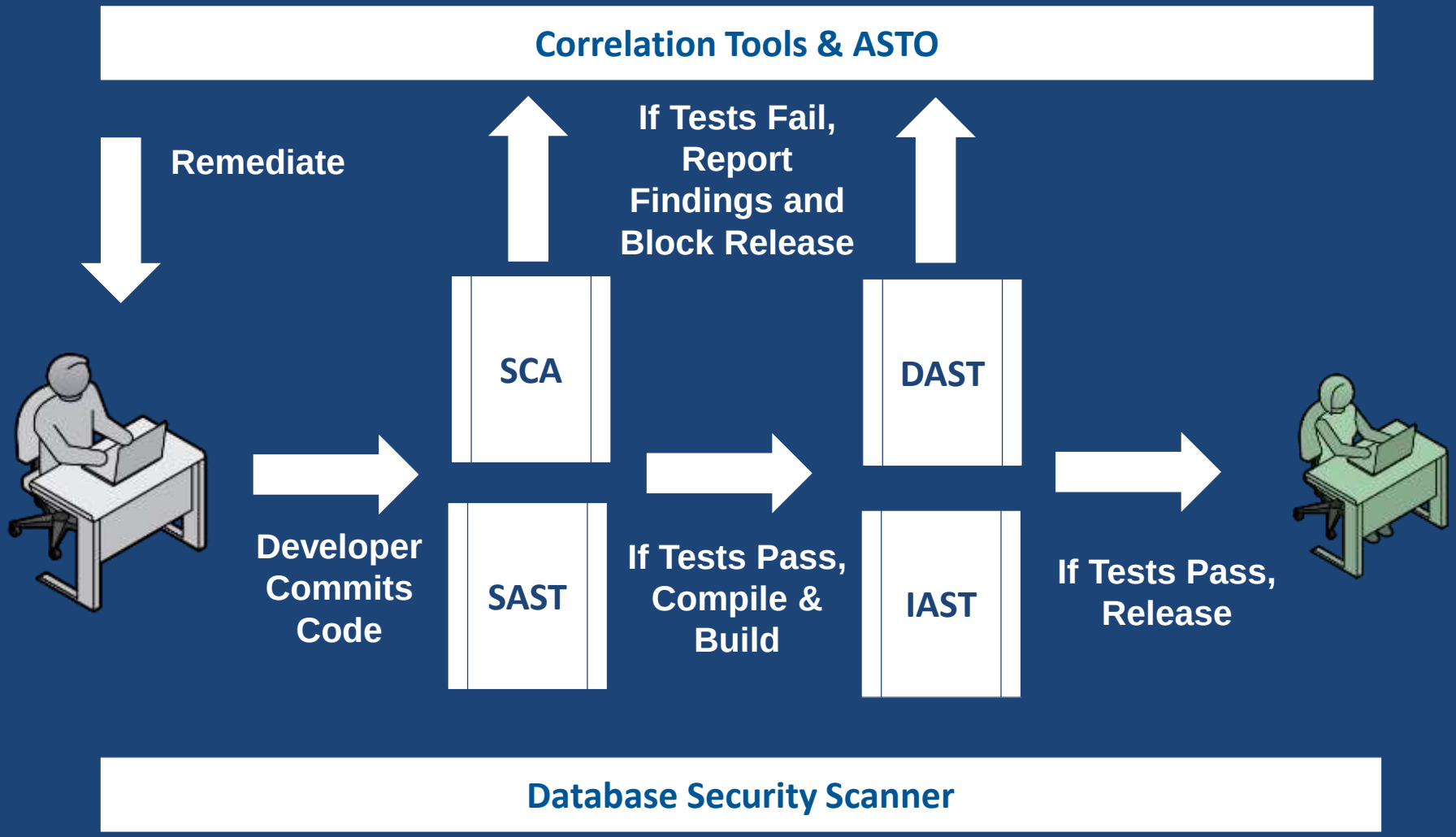
- SCA tools examine software to determine the origins of all components and libraries
- Most effective finding common and popular libraries and components, particularly open source pieces
- Compares modules found in code to list of known vulnerabilities
- Almost all use NIST National Vulnerability Database 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

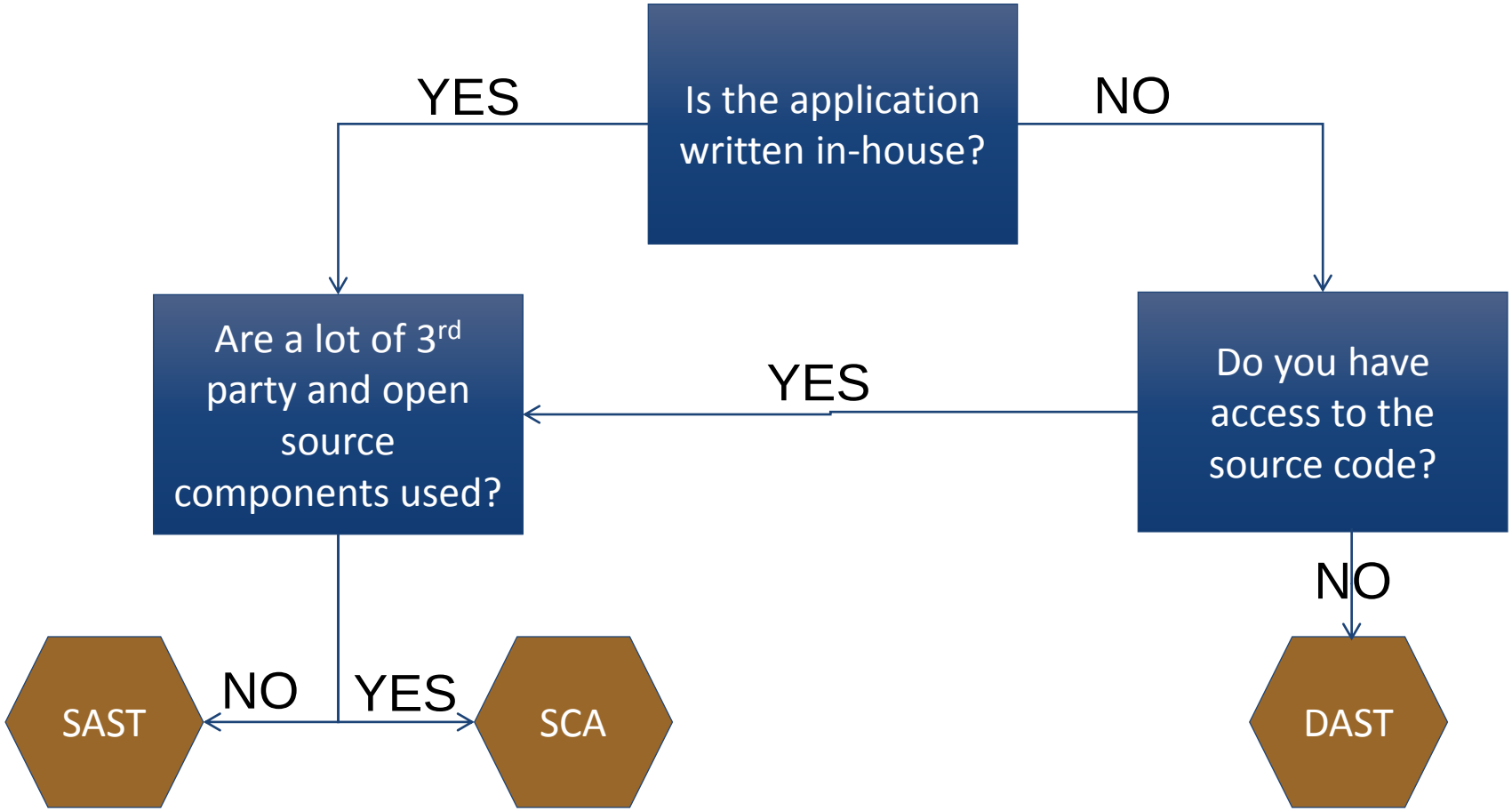


Application Security Testing Tools Reference Model CI/CD Development Project

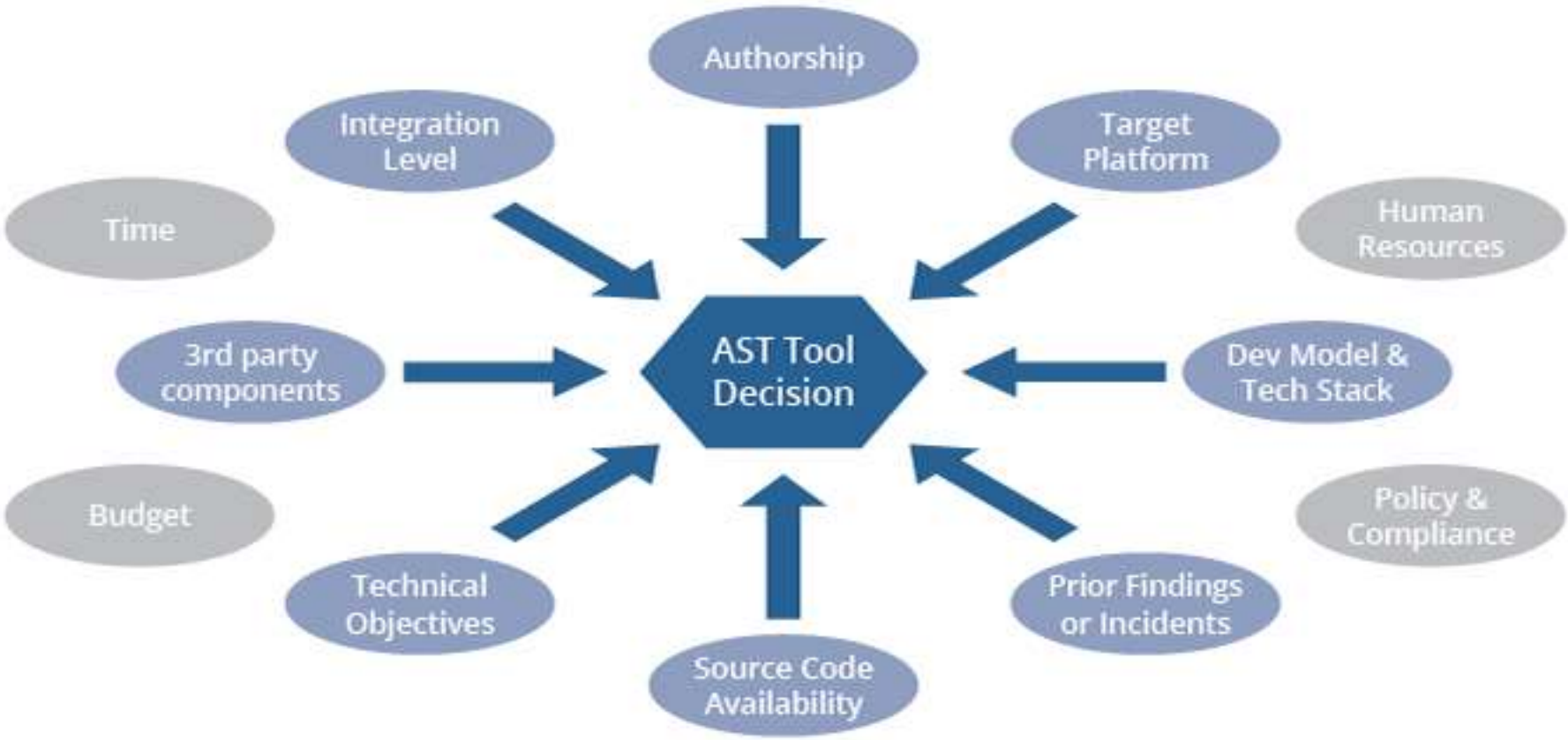


A Simple AST Decision Flow

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.**