There Is No Plug & Play: Tips for Implementing Automated Software Security Testing Tools

Dr. Thomas P. Scanlon

Security Automation Directorate
CERT Division
Software Engineering Institute
Carnegie Mellon University
Pittsburgh, PA 15213

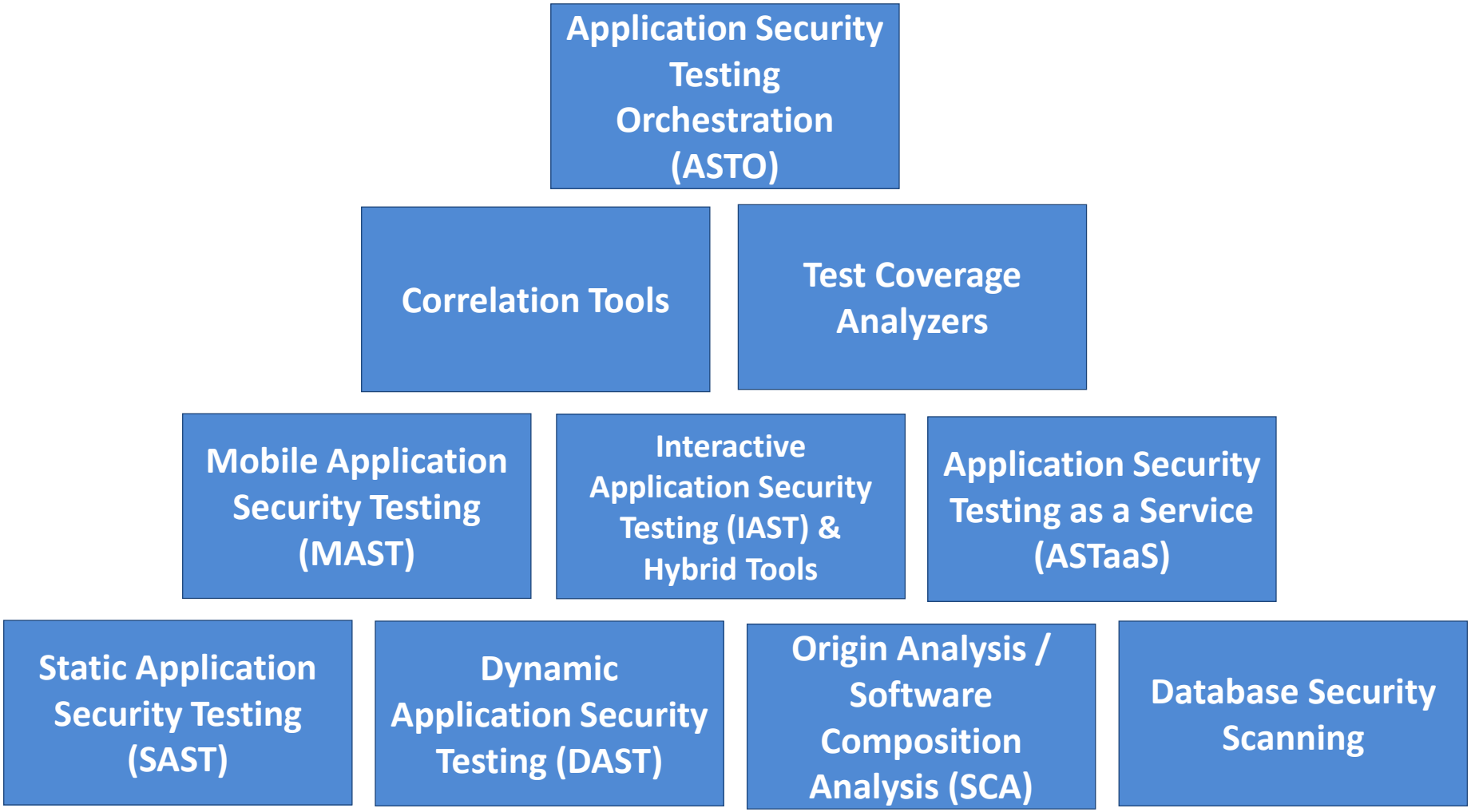
Document Markings

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.

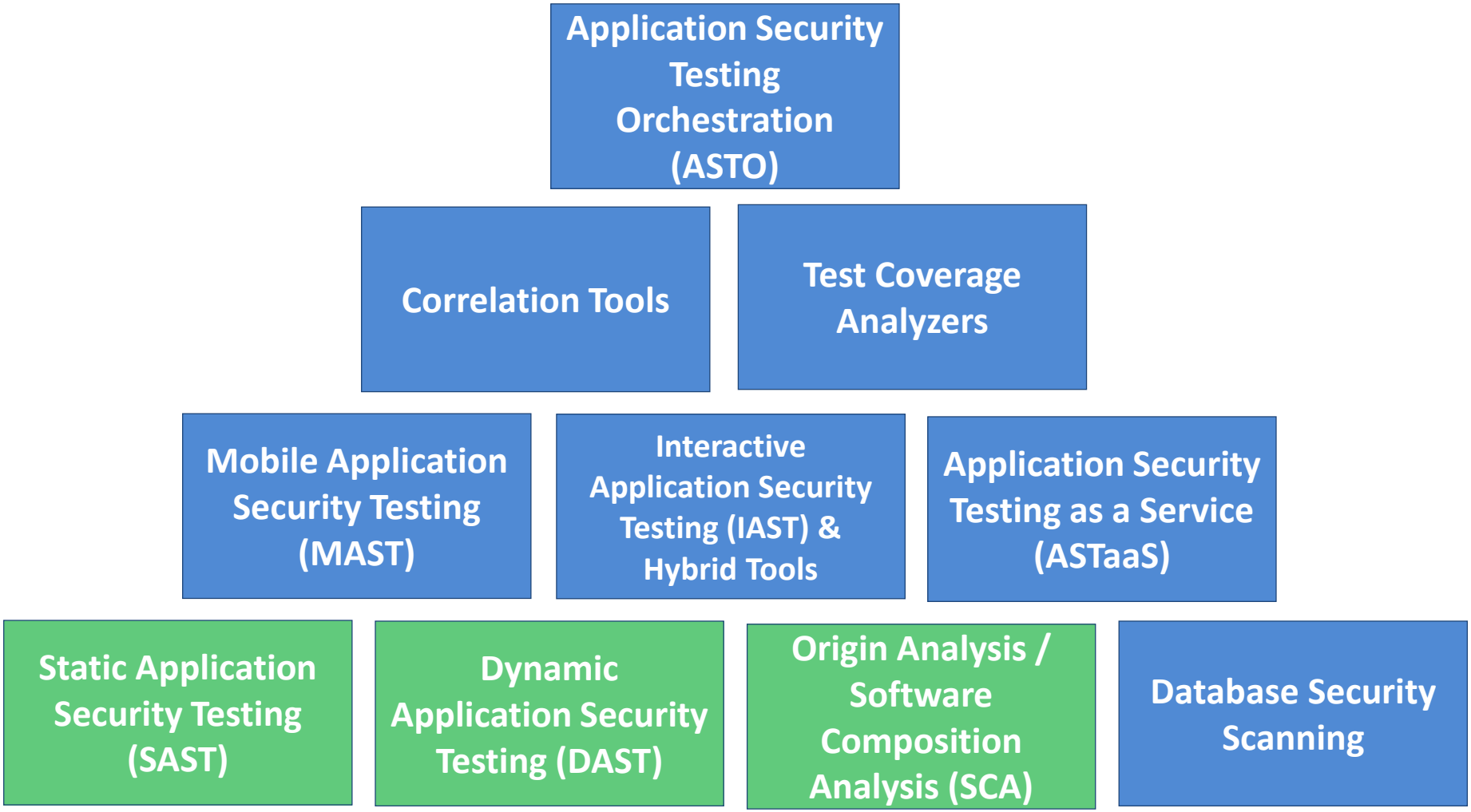
DM18-1076

Application Security Testing (AST) Tools Overview

Application Security Testing Tools Pyramid



Application Security Testing Tools Pyramid



There are a lot of good reasons to use these tools…

- All software has bugs and weaknesses
- Manual code reviews and traditional test plans are very time consuming
- Increase speed, efficiency, and coverage paths
- Repeatable
- Scale well
- Find known vulnerabilities, issues and weaknesses
- New vulnerabilities are continually being introduced and/or discovered
- Regulatory and compliance directives
- The bad guys use tools too!!

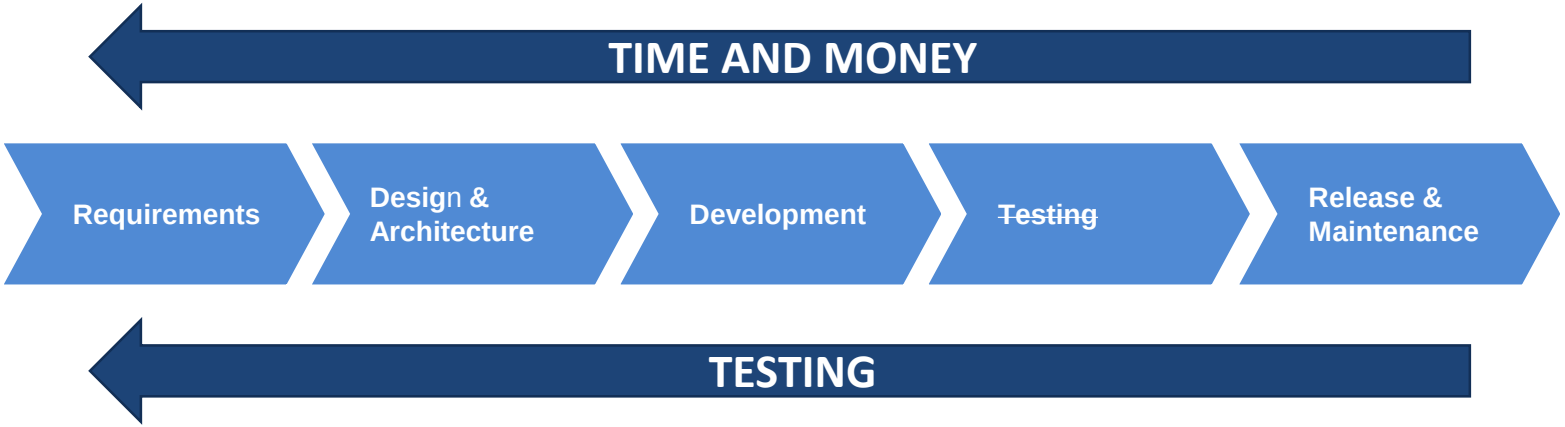
…but it's usually not as easy as "plug and play"

Implementation Challenge: Not Enough Resources

The Tools Require Resources to Integrate & Operate

Do “shift left” for testing

But also “shift left” for budgeting and scheduling



Make Sure You Have Sufficient Infrastructure

- **In some cases, different hardware configurations can impact dynamic code analysis tools**
- **Will users running different hardware & configurations find issues that didn't occur in DAST testing?**
- **Can't test on all platforms realistically (time/money), but be aware of potential impacts**

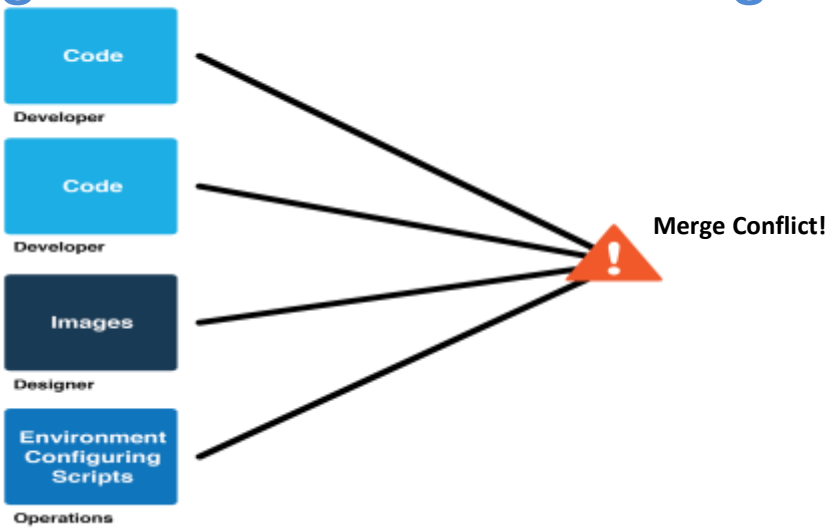
Make Sure You Have Sufficient Infrastructure

- **Software composition analysis tools can be very "noisy" on the network – they need to phone home**
- **If running SCA tools in a disconnected environment, you will need to store a lot data**
- **If your internal networks are isolated, need multiple copies of same large database**

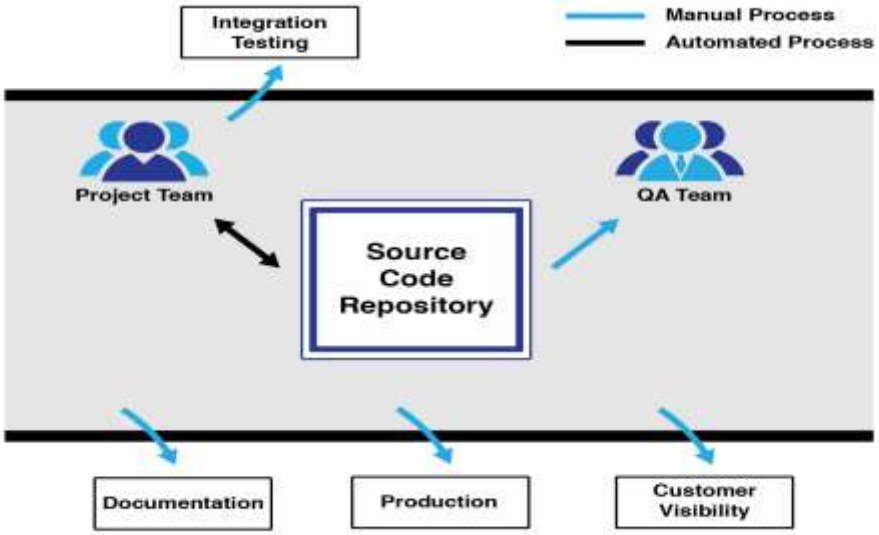
Implementation Challenge: Integration

Software projects consist of many artifacts

Integration can be challenging



This is often a manual process



Manual Integration is a Burden

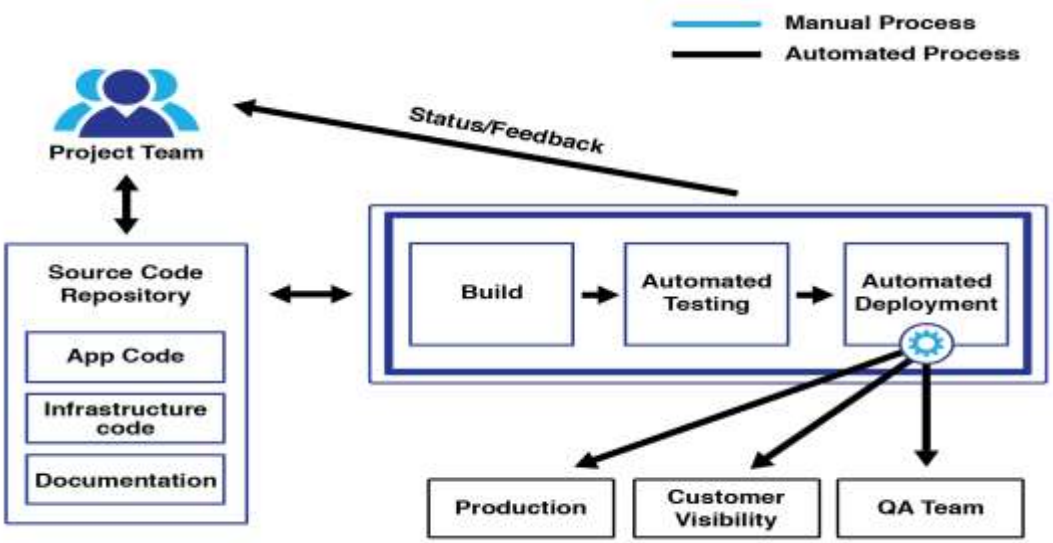
Human-driven processes are…

- **Infrequent**
- **Expensive**
- **Repetitive**
- **Error-prone**

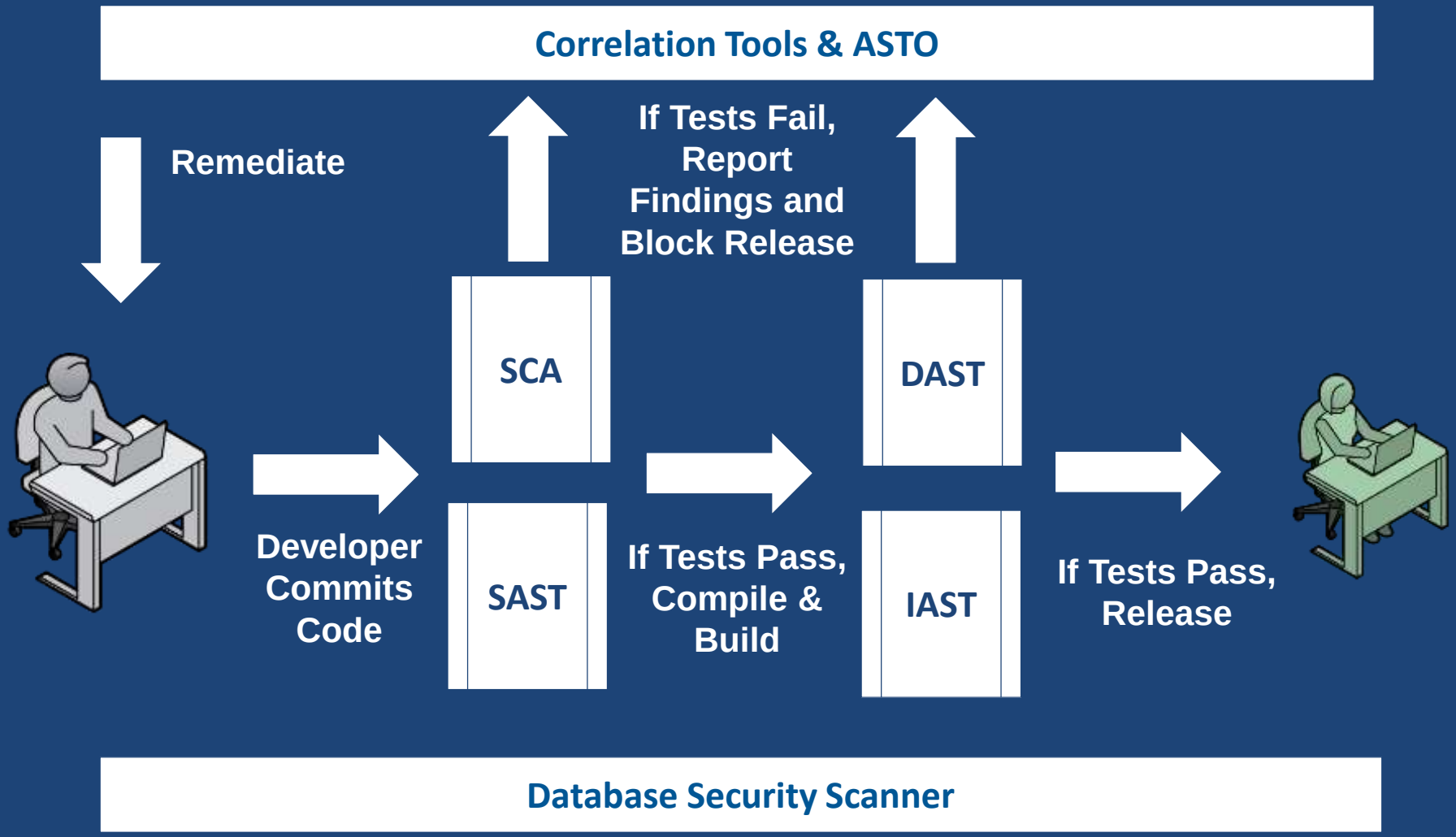
This leads to:

- **Disjointed** activities / components
- **Slow**, unreliable, costly reporting and failure recognition
- **Lack of transparency** of problems
- **Integration Nightmare**

Highly Automated Continuous Integration (CI) Reference Model



Application Security Testing Tools Reference Model CI/CD Development Project



Automating Integration Provides Benefits

Automation…

Removes inefficiencies due to human-driven process

Standardizes artifact submission process

Guarantees **consistent results**

Allows team to **fail fast** (and fix fast)

Reduces pain of integration

Continuous Integration Provides Even More Benefits

Continuous Integration uses a **build server** to…

Integrate artifacts on **every change**

Give team with **immediate notification** of failure or success

Require issues be fixed before moving forward

Enforce standards (can fail based on security controls as well as functionality)

Continuous Integration Requires Some Discipline

For successful implementation of a CI process:

Developers must **commit changes often**

CI system should **build every commit**

Automate every step of the build process

Automate tests, and fail the build on test failure

CI system should **report results immediately** to everyone

CI system should **instantly revert** to previous release on failure

All environments should have **100% parity**

Integration Considerations

- **The tools will need to be integrated into your technical ecosystem – allow time & effort for this**
- **Make sure your tool is compatible with your ecosystem**
- **Make sure your tool supports your target programming languages (yeah really)**
- **Make sure the version of the tool you are using supports the version of languages and OS you need**
- **Some of the tools are known to abandon older versions of languages/OS relatively quickly**

Implementation Challenge: Dealing with Tool Output

These Tools Generate A LOT of Output

Cppcheck report - LibreOffice 2018-09-06 22:34:57 260740f0a08fcefb9a1c7d43e775e344064f2af9, CppCheck 2018-09-08 14:10:34 fa40b821e652b32a2122f0e088ede8f1ec203b1b:

		Line	Id	CWE	Severity	Message
2	nullPointer					
2	returnTempReference		missingInclude		information	Cppcheck cannot find all the include files (use --check-config for details)
2	unsafeClassCanLeak	UnoControls/source/base/registercontrols.cxx				
2	uselessAssignmentAr	78	unusedFunction	561	style	The function 'ctl_component_getFactory' is never used.
2	wrongPrintfScanfArgN	accessibility/source/extended/accessibletabbarbase.cxx				
1	accessForwarded	42	syntaxError		error	syntax error
1	accessMoved	accessibility/source/extended/accessibletablistboxtable.cxx				
1	arrayIndexThenCheck	255	syntaxError		error	syntax error
1	asctimeCalled	accessibility/source/extended/listboxaccessible.cxx				
1	class_X_Y	44	syntaxError		error	syntax error
1	constStatement	accessibility/source/extended/textwindowaccessibility.cxx				
1	doubleFree	1530	syntaxError		error	syntax error
1	duplicateBreak	accessibility/source/standard/vclxaccessiblebutton.cxx				
1	exceptThrowInDestru	262	knownConditionTrueFalse	570	style	Condition 'nValue>1' is always false
1	invalidFunctionArg	accessibility/source/standard/vclxaccessibleradiobutton.cxx				
1	memsetClass	235	knownConditionTrueFalse	570	style	Condition 'nValue>1' is always false
1	missingInclude	accessibility/source/standard/vclxaccessiblescrollbar.cxx				
1	negativeContainerInd	205	knownConditionTrueFalse	570	style	Condition 'nValue>nValueMax' is always false
1	noDestructor	accessibility/source/standard/vclxaccessibletoolboxitem.cxx				
1	nullPointerArithmeticF	702	knownConditionTrueFalse	570	style	Condition 'nValue>1' is always false
1	pointerSize	animations/source/animcore/animcore.cxx				
1	redundantCondition	1552	unusedFunction	561	style	The function 'setOrigin' is never used.
1	stlIfStrFind	1783	unusedFunction	561	style	The function 'insertAfter' is never used.
1	uninitMemberVarPriva	avmedia/source/avmediadummy.cxx				
1	zerodivcond	98	unusedFunction	561	style	The function 'com_sun_star_comp_framework_SoundHandler_get_implementation' is never used.
9003 total		avmedia/source/framework/mediacontrol.cxx				
Statistics		261	syntaxError		error	syntax error
		avmedia/source/framework/soundhandler.cxx				
		299	syntaxError		error	syntax error
		avmedia/source/gstreamer/gstframegrabber.cxx				
		166	redundantAssignment	563	style	Variable 'pData' is reassigned a value before the old one has been used.
		166	redundantAssignment	563	style	Variable 'pData' is reassigned a value before the old one has been used.
		avmedia/source/gstreamer/gstplayer.cxx				
		239	syntaxError		error	syntax error
		239	syntaxError		error	syntax error
		avmedia/source/gstreamer/gstuno.cxx				
		42	unusedFunction	561	style	The function 'avmediagst_component_getFactory' is never used.
		avmedia/source/viewer/mediawindow_impl.cxx				

Allocate Resources to Deal with the Output

- Can you use the ”out of the box” settings? Most likely not, will need to calibrate the knobs and buttons.
- Know the difference between warnings, errors, syntax, performance, information, and style issues.
- A lot of issues discovered are more stylistic and do not necessarily impact security, behavior, or performance.
- Investigating findings can’t just be done from the report, it involves a lot of digging into the code and running tests.
- But be careful, everything is not always what it seems, hence the resource-intensive inspection

Taking a Closer Look – Basic Tuning

These style issues can possibly be suppressed; they are interesting and may point to some inefficiency, but are possibly low priority

1843	redundantAssignment	563	style	Variable 'pNewLeft' is reassigned a value before the old one has been used.
1844	redundantAssignment	563	style	Variable 'pNewRight' is reassigned a value before the old one has been used.
1862	redundantAssignment	563	style	Variable 'pNewLeft' is reassigned a value before the old one has been used.
1863	redundantAssignment	563	style	Variable 'pNewRight' is reassigned a value before the old one has been used.

Taking a Closer Look – Basic Tuning

These syntax errors are probably a higher priority – they may cause breaking behavior or worse

dbaccess/source/ui/dlg/advancedsettings.cxx			
131	syntaxError	error	syntax error
dbaccess/source/ui/dlg/dbfindex.cxx			
171	syntaxError	error	syntax error
dbaccess/source/ui/dlg/dbwiz.cxx			
110	syntaxError	error	syntax error
dbaccess/source/ui/dlg/dbwizsetup.cxx			
272	syntaxError	error	syntax error

Things Aren't Always What They Seem To Be

These are labeled "style", but a bunch of functions that aren't called can be a security issue

136	unusedFunction	561	style	The function 'pasteFormat' is never used.
159	unusedFunction	561	style	The function 'openDialog' is never used.
296	unusedFunction	561	style	The function 'getApplicationMainWindow' is never used.
485	unusedFunction	561	style	The function 'previewChanged' is never used.
512	unusedFunction	561	style	The function 'askToReconnect' is never used.
605	unusedFunction	561	style	The function 'isRenameDeleteAllowed' is never used.

Implementation Challenge:

Knowing How Much To Automate

Fail the Build When Software isn't Good Enough

Don't just configure failure for compile/build errors!

- Does the changes include a know weak coding practices (CWE)?
  - Automatically run changes against a static code analysis tool and fail the build if a new CWE is found
- Do any of the current or new libraries have known vulnerabilities (CVE)?
- Did any Functional Security Tests Fail?
- Example NIST RMF Security Controls:
  - SA-11 Developer Security Testing and Evaluation
  - SA-12 Supply Chain Protection
  - CM-4 Security Impact Analysis
  - RA-3 Risk Assessment

CI is your best tool to enforce security standards

But How Much Automation Is Appropriate?

- **Will you allow tool findings to break (stop) builds?**
- **At first pass, many people want tools findings to break builds...until it happens too much.**
- **You will need to develop a process to prioritize and triage findings and establish a severity threshold for what are showstoppers.**
- **Most of the tools support this, but they require configuration and tuning – which requires resources.**

But How Much Automation Is Appropriate?

- If the tools aren't allowed to break builds, you need to establish policies and procedures for addressing findings so that they aren't just ignored.
- Automated tools may require ACL and privileges changes, which may also require policy changes (multi-factor authentication, smart cards, password storing, etc)

Summary of Implementation Challenges

- **Enough Resources Aren't Allocated (#1 Reason!)**
- **There will be integration work to get things up and running smoothly, make sure to plan and budget for it**
- **You will need to make sure the tools fit into your technical ecosystem (languages, OS, versions)**
- **Anticipate a lot of tool output, have a plan for prioritizing, triaging, and processing it**
- **Develop a strategy for how the tool findings will impact build processes and automation**
- **Some tools and test scenarios require more infrastructure than it may seem, be prepared for this**

Post-Implementation Challenges

Tools Are Up and Running: Now What?

- **The tools themselves will need care and feeding – new tool versions and patches are regularly released**
- **SCA tools in particular will continue to necessitate a lot of network traffic to keep up with emerging threats**
- **If you installed a SCA database locally, you will need a strategy to synch it with current ("sneakernet")**

Business Rules and Legal Issues

- SCA tools can enforce business rules, compliance regulations, software licensing issues, supply chain, rules and more, but this requires custom configuration
- The input for these configurations need more than just input from IT folks (management, legal, auditing, etc.)
- Licensing can be a very complicated topic. It's not just the licenses you inherit from components you integrate, it's also the licenses those components inherit
- What do you if a SCA tool finds a license conflict in a product you already have released?

False Sense of Security?

- **Static code analysis tools traditionally had a high rate of false positives – they were identifying issues that weren't valid**
- **Consumers of these tools complained a lot about false positives occupying their resources**
- **More recently, static code analyzers have become much, much better at reducing false positives**
- **But at what cost?**

A Quick Detour on Precision and Recall

Precision is a measure of the issues found that are legitimate issues (true positives) in relation to the amount of issues found that are not valid (false positives)

$$\text{Precision} = \frac{tp}{tp + fp}$$

Recall is a measure of the legitimate issues (true positives) found in relation in relation to the total universe of legitimate issues that exist

$$\text{Recall} = \frac{tp}{tp + fn}$$

SAST Precision and Recall

- **Many static code analysis tools now have a high rate of precision – the issues they return are almost always valid and contain very few false positives**
- **However, many static code analysis tools have low recall rates - they aren't finding many of the issues present in the software**
- **This means that while the tool output can be trusted as valid items to remediate, there are a lot of issues in code bases that they aren't detecting**

More False Sense of Security?

- **Dynamic code analysis tools traditionally have less False Positives due to the nature of the tools (they run until they find an issue – often no disputing the issue occurred)**
- **BUT…were all the branches/path routes covered? Only the errors that occur in the path actually executed can be detected**
- **SCA Tools only find known vulnerabilities, more specifically, known vulnerabilities that are in there database**

The Takeaway on AST Tools Findings

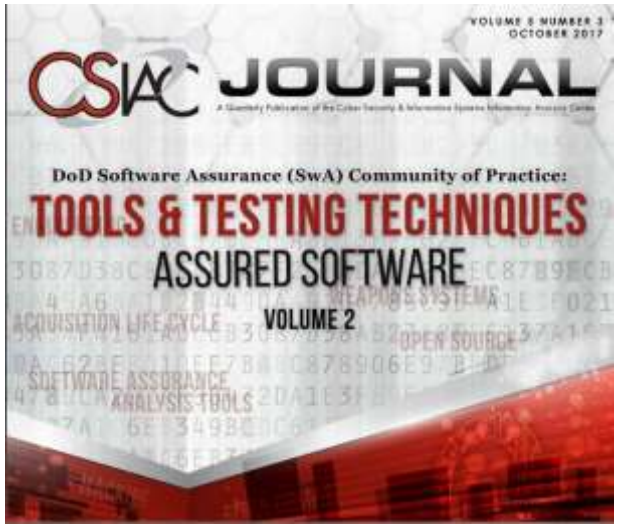
- **SAST, DAST, and SCA tools should most definitely be used as part of a comprehensive application security program**
- **The tools are not one-stop shops for application security – you can't simply run the tools and believe you are secure as long as the tool reports are favorable**

A Case Study:
The Role People Play
In Tools' Success

People Make The Tools Work

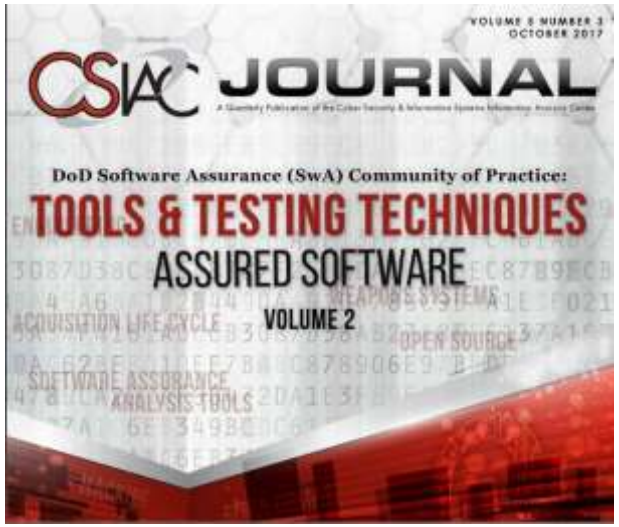
- **Automated testing tools require human resources to install, configure, and maintain tools as well as investigate and remediate findings**
- **But people also drive the overall success of the tools in the project and organizations**
- **It is the subjective opinions on tool effectiveness, utility, and usability that will determine successful tool adoption**

Case Study: Perceptions on Tools



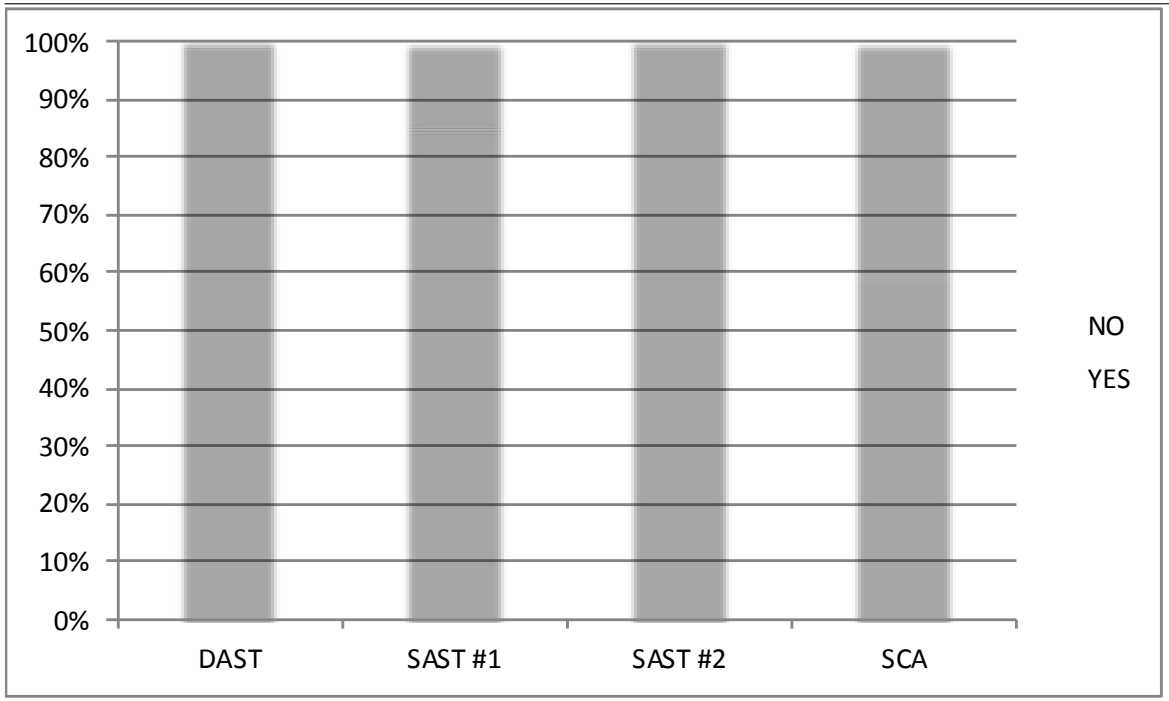
- 66 different programs / projects received licenses through a pilot AST tools program
- Dynamic web testing tool (DAST)
- Two static source code analyzers (SAST)
- Origin analysis tool (SCA)
- As part of the agreement to use the software licenses, pilot participants had to agree to complete a survey at the end of the license year.

Case Study: Perceptions on Tools



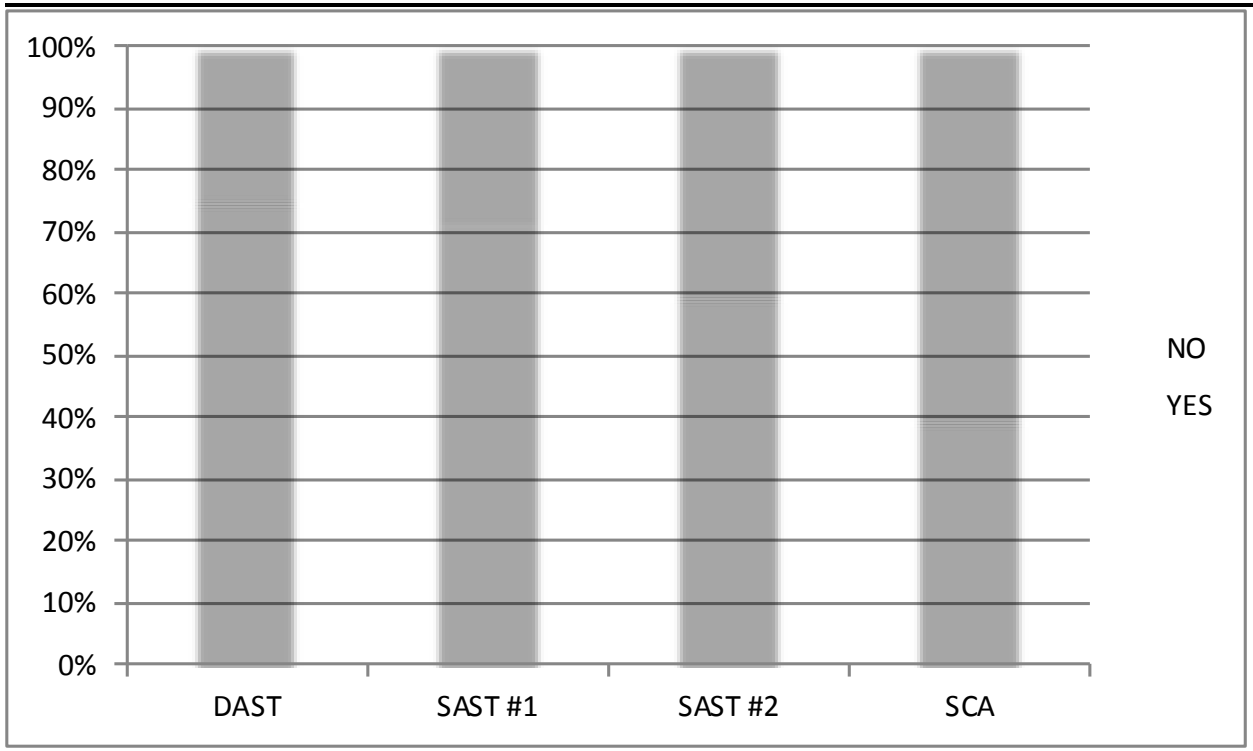
- 248 licenses were distributed across the four product offerings
- 9,000+ projects scanned
- 49,000,000+ lines of code scanned
- 860,000+ issues detected

Did the tools find good things or a lot of noise?



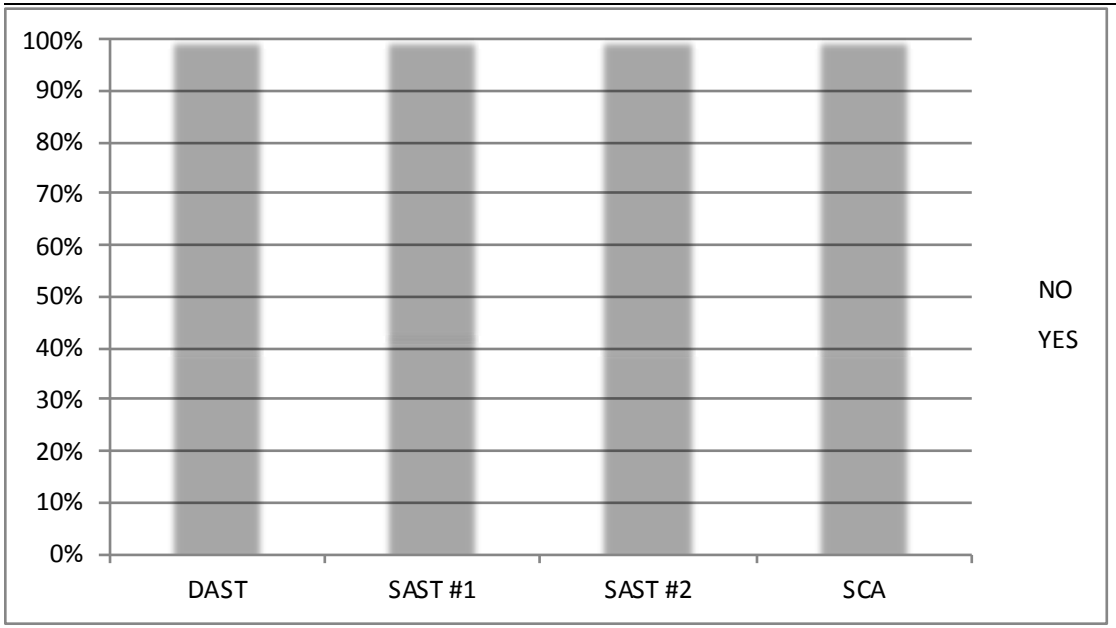
Did the tool find meaningful issues that need to be addressed?

But, did it find really good things?



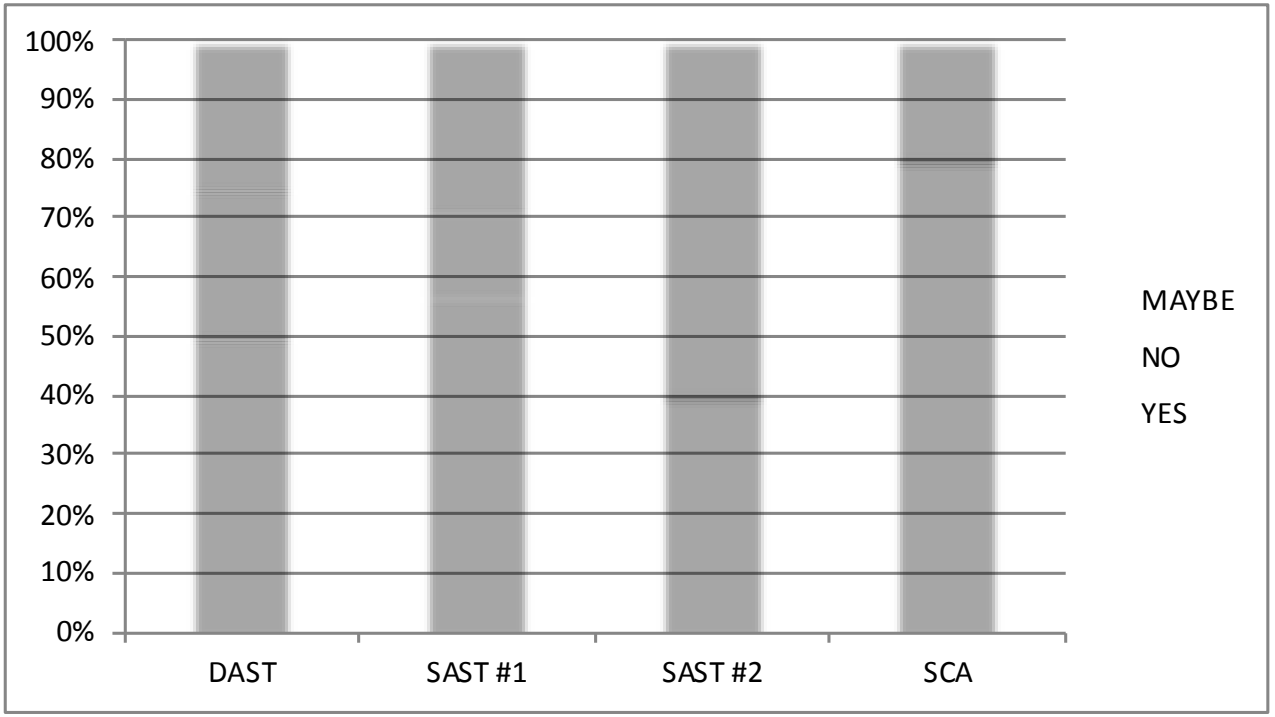
Did the tool find issues that you felt required IMMEDIATE attention?

So, you fixed these things right?



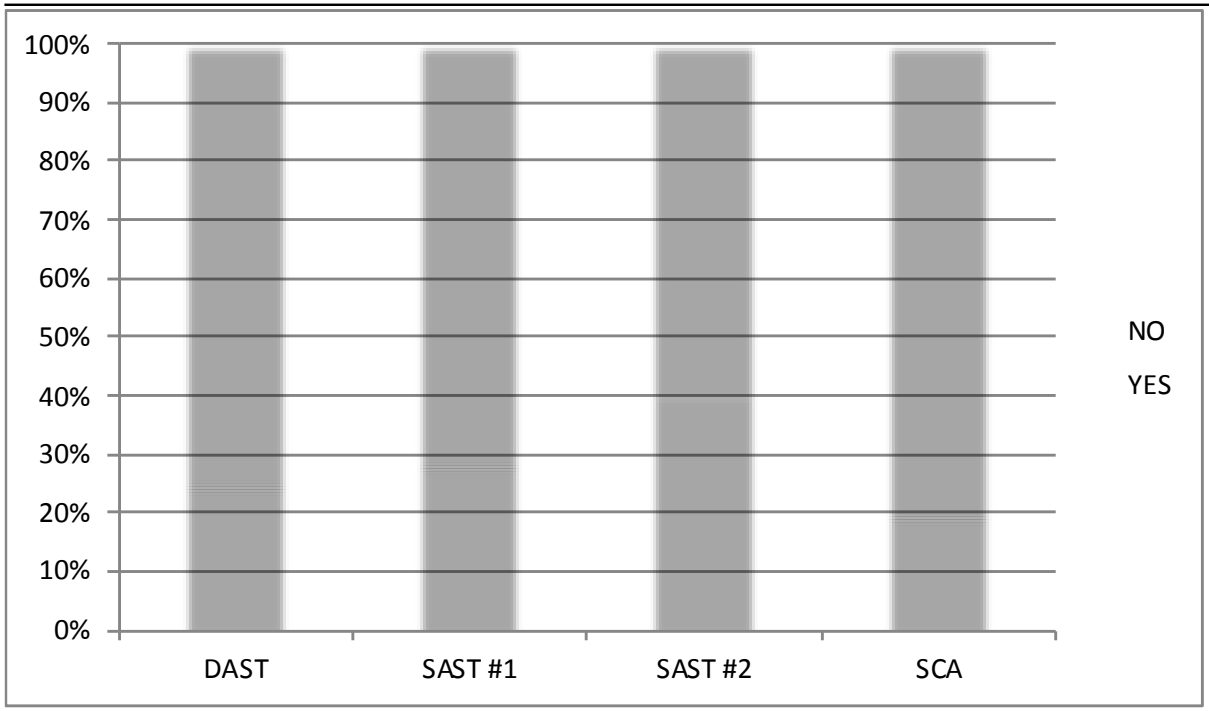
To date, have you fixed/addressed any issues, warnings, and/or vulnerabilities as a result of the tool feedback?

But you promise to fix them, right?



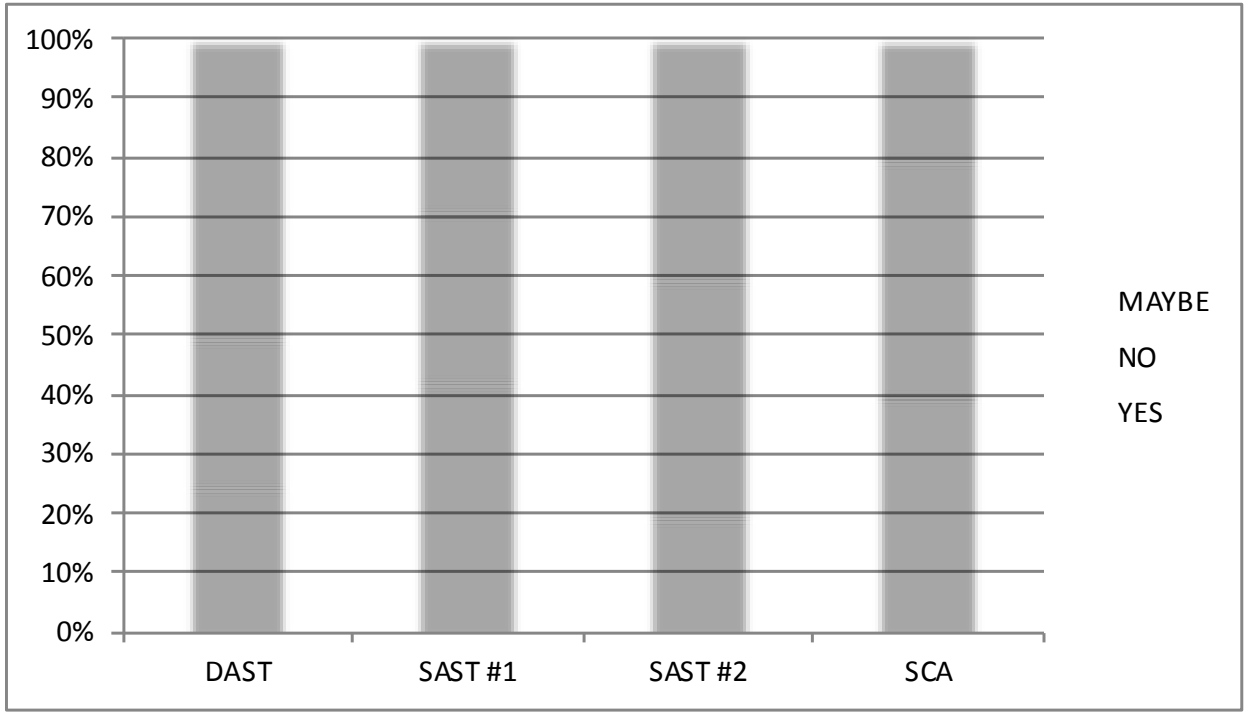
Are there future plans to fix or address any issues, warnings, and/or vulnerabilities as a result of the tool feedback?

At least, you'll stop introducing these issues?



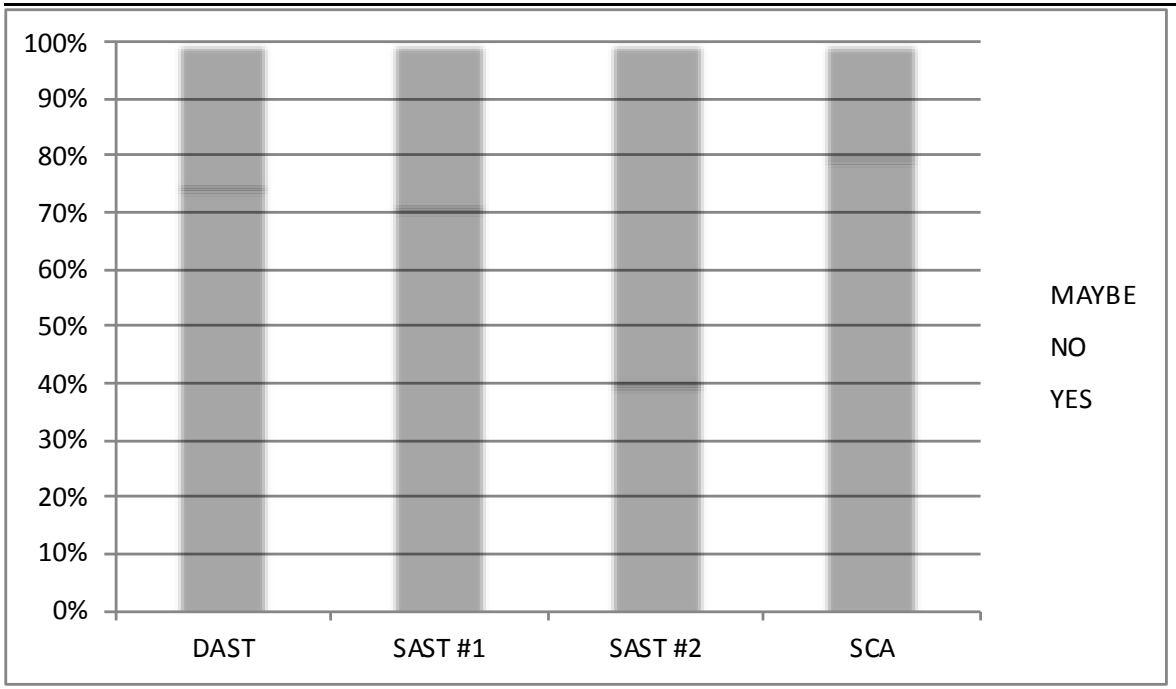
Have you made changes to your design, development, or build processes as result of having used this tool?

Maybe later?



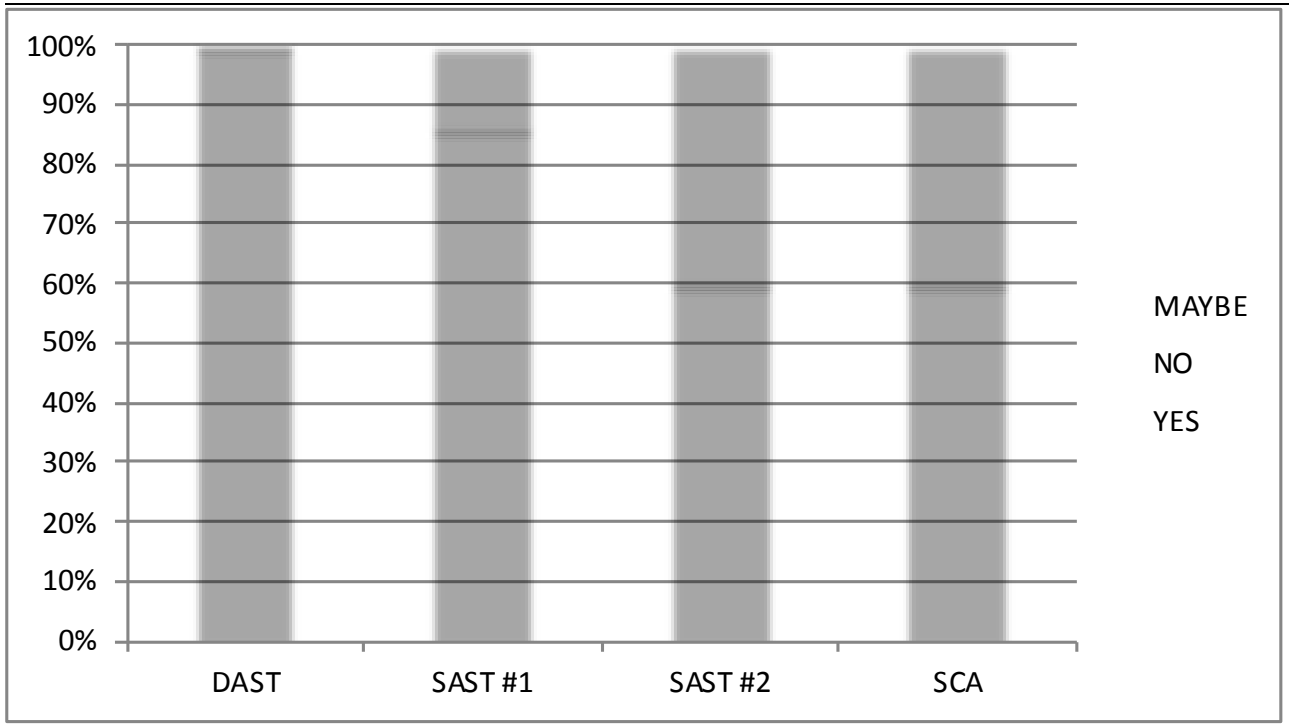
Are there future plans to make changes to your design, development, or build processes as result of having used this tool?

Would this tool be helpful to your process?



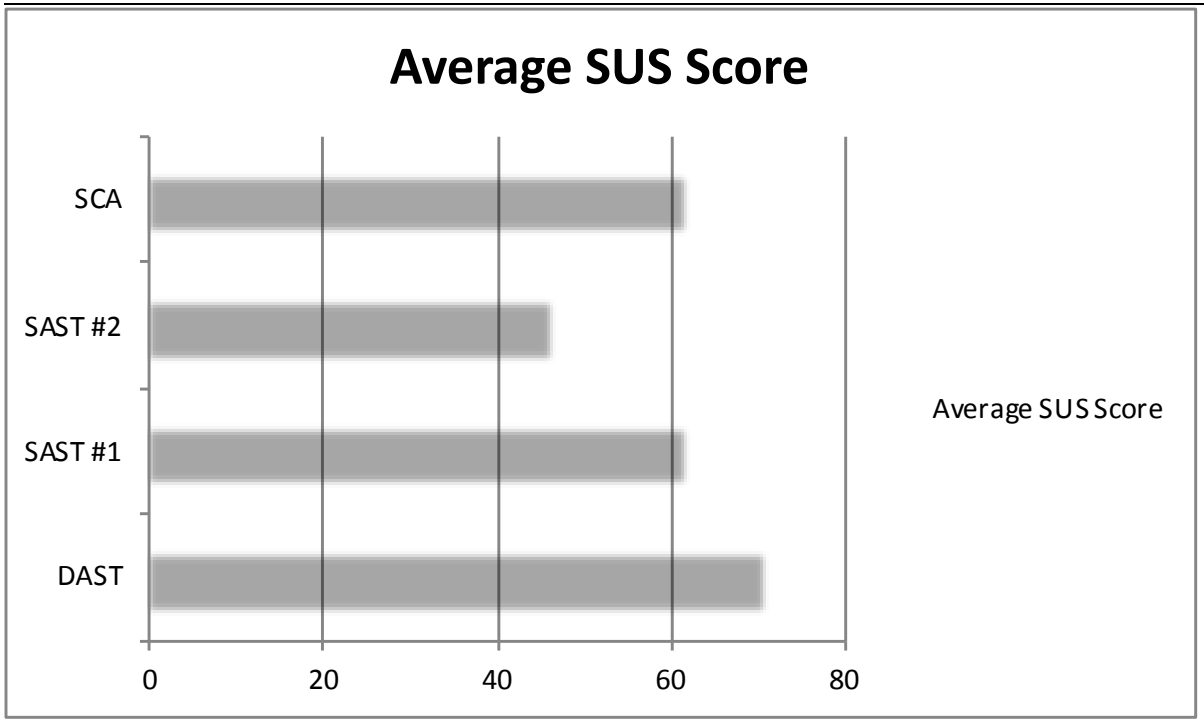
Do you think this tool is effective in finding meaningful issues, warnings, and/or vulnerabilities in your projects and applications?

Would you like to keep these tools in your process?



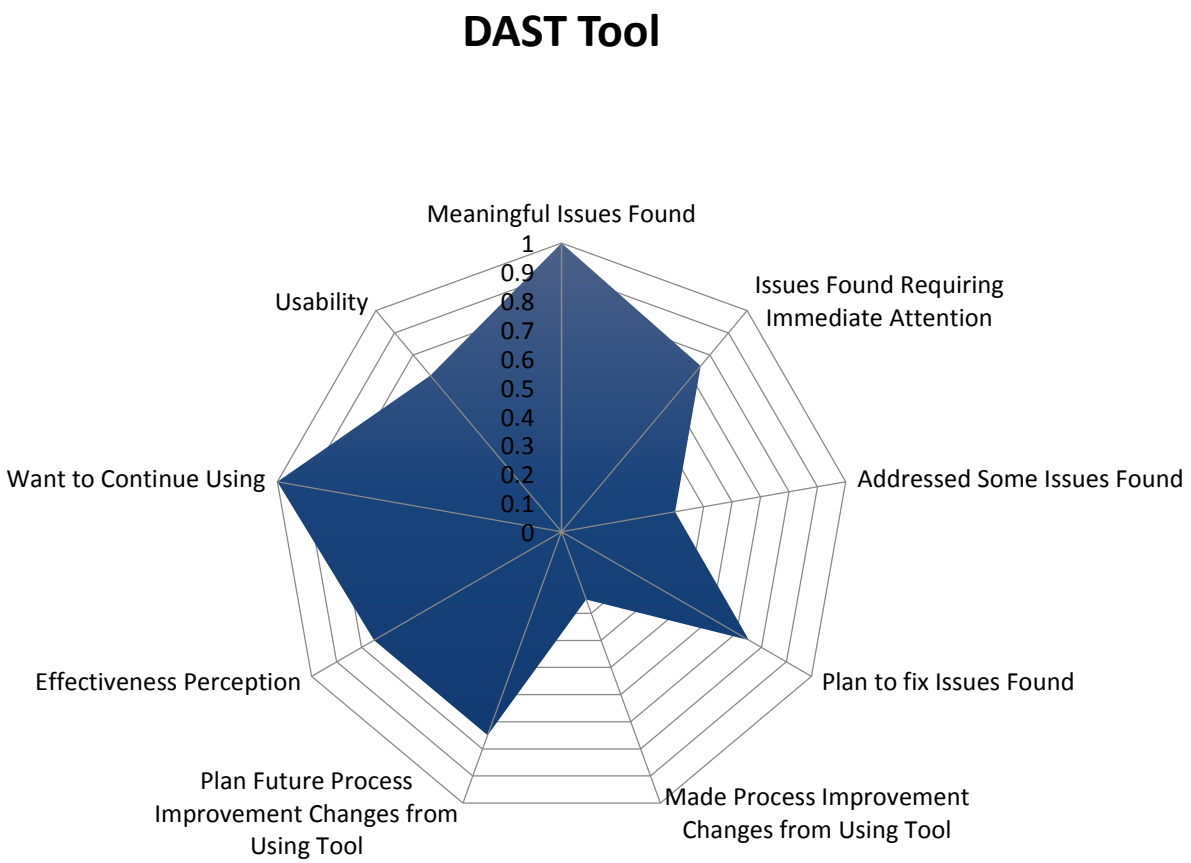
Would you like to continue using this tool in your projects and applications?

How “usable” were these tools?

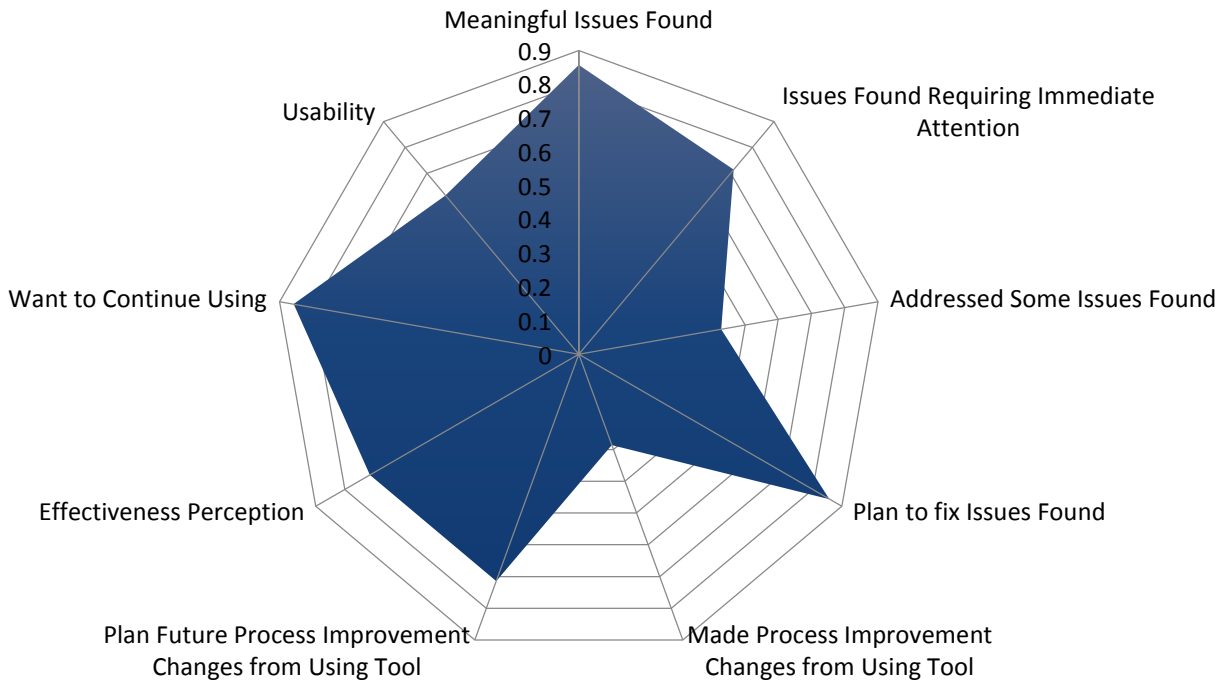


Average SUS usability score for each tool

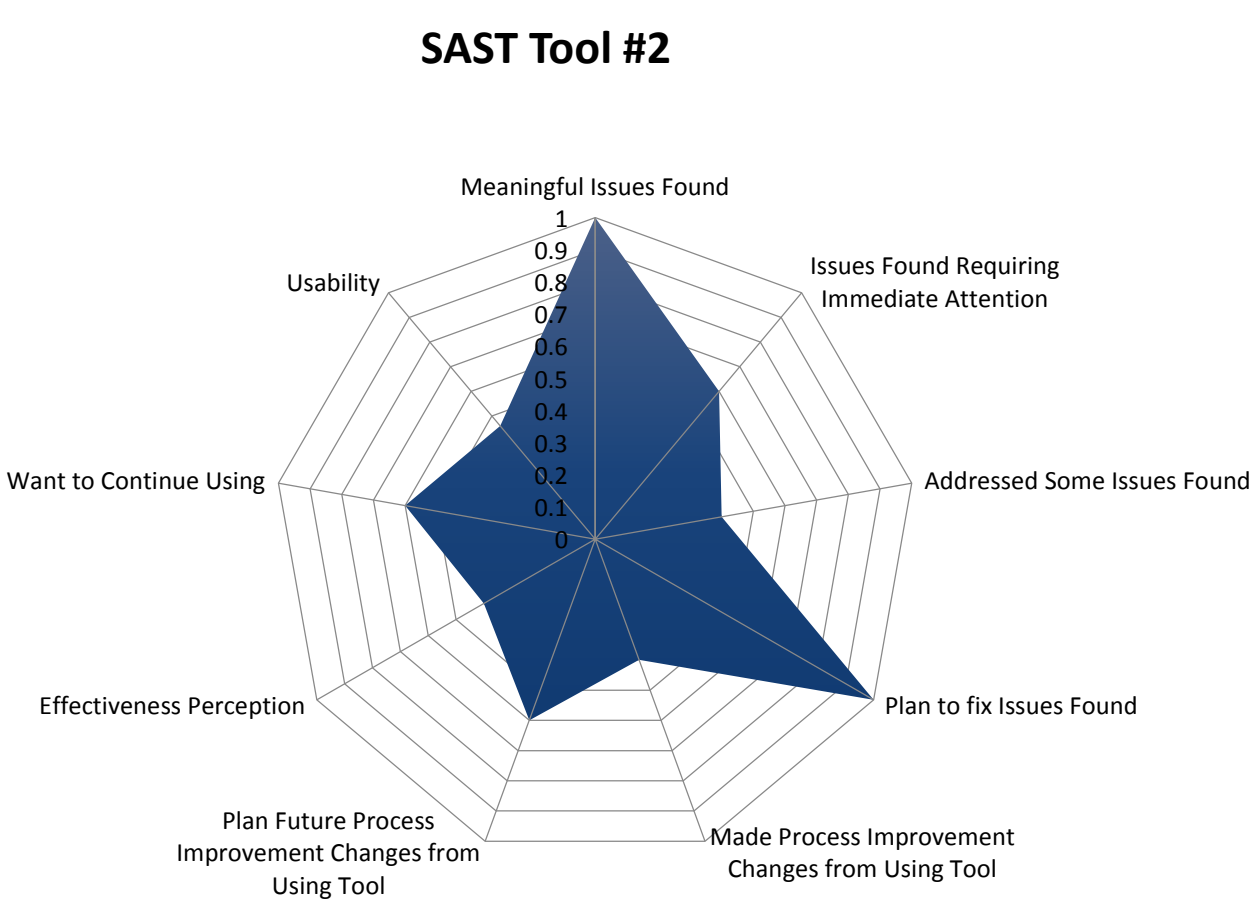
DAST Summary



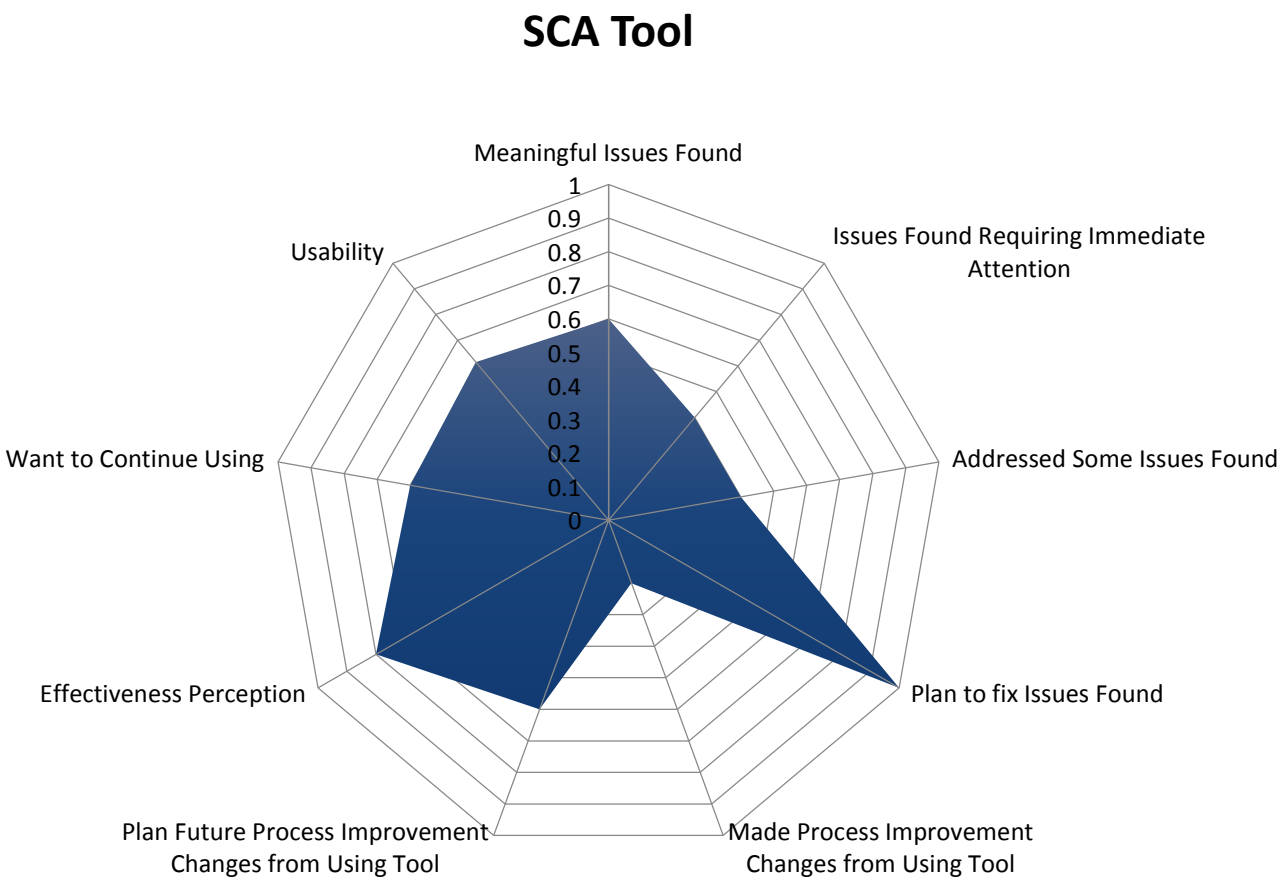
SAST #1 Summary



SAST #2 Summary



SCA Summary



Resources for Information In This Talk

SEI Blog

10 Types of Application Security Testing Tools: When and How to Use Them

https://insights.sei.cmu.edu/sei_blog/2018/07/10-types-of-application-security-testing-tools-when-and-how-to-use-them.html

SEI Blog

Decision-Making Factors for Selecting Application Security Testing Tools

https://insights.sei.cmu.edu/sei_blog/2018/08/decision-making-factors-for-selecting-application-security-testing-tools.html

CSIAC Journal

Piloting Software Assurance Tools in the Department of Defense

https://www.csiac.org/journal-article/piloting-software-assurance-tools-in-the-department-of-defense/

AST Tool Resources

- NIST Samate Project: https://samate.nist.gov/Main_Page.html
- OWASP: https://www.owasp.org/index.php/Main_Page
- SANS: https://www.sans.org/about/
- SEI: https://www.sei.cmu.edu/

  http://www.cert.org/secure-coding/tools/index.cfm
- DHS Build Security In: https://www.us-cert.gov/bsi
- GNU Hurd: https://www.gnu.org/software/hurd/open_issues/code_analysis.html

Contact Information

Dr. Thomas P. Scanlon
Cybersecurity Researcher
CERT Security Automation Directorate
Software Engineering Institute
Carnegie Mellon University
Telephone: +1 412.268.9209
Email: scanlon@cert.org