Automating application security testing… and shifting it left

Frans van Buul, Micro Focus

About me

- Presales for the Micro Focus Fortify application security testing portfolio, since 2014.
- Based in the Netherlands, leading the Fortify presales practice across EMEA and LATAM.
- Background in security consulting/auditing and (Java) software development.

- Contact me: frans.buul@microfocus.com

Agenda

- Introduction to Application Security – what is and why care?
- Testing for application security – techniques and challenges
- Shifting left application security

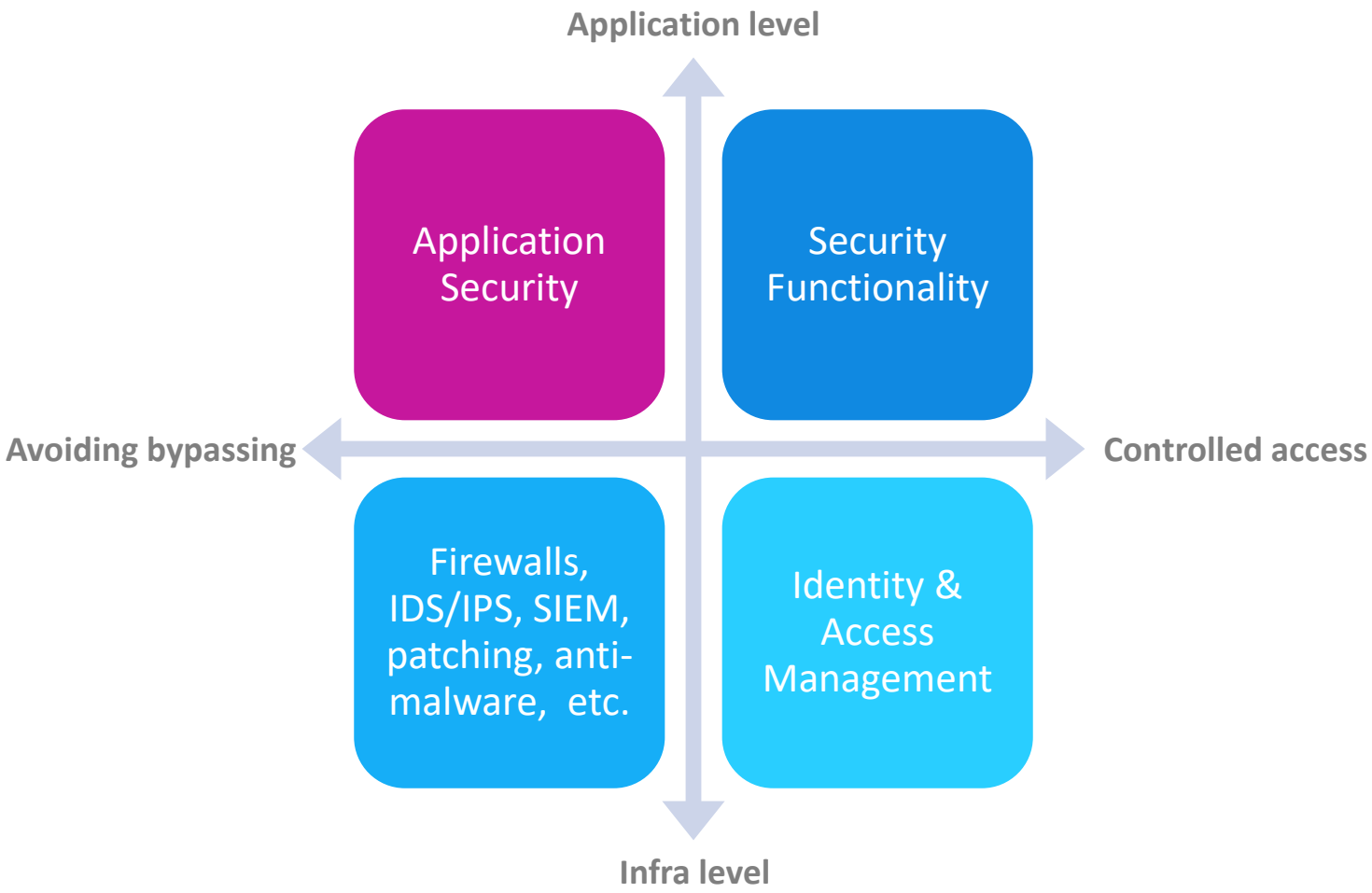
Want to learn about the great Fortify products for all of this?

Come to our booth, drop me an email, or visit
https://www.microfocus.com/en-us/solutions/application-security

Introduction to Application Security – what is and why care?


The security of controlled access
Required for
The security of not bypassing security functionality.

A security quadrant

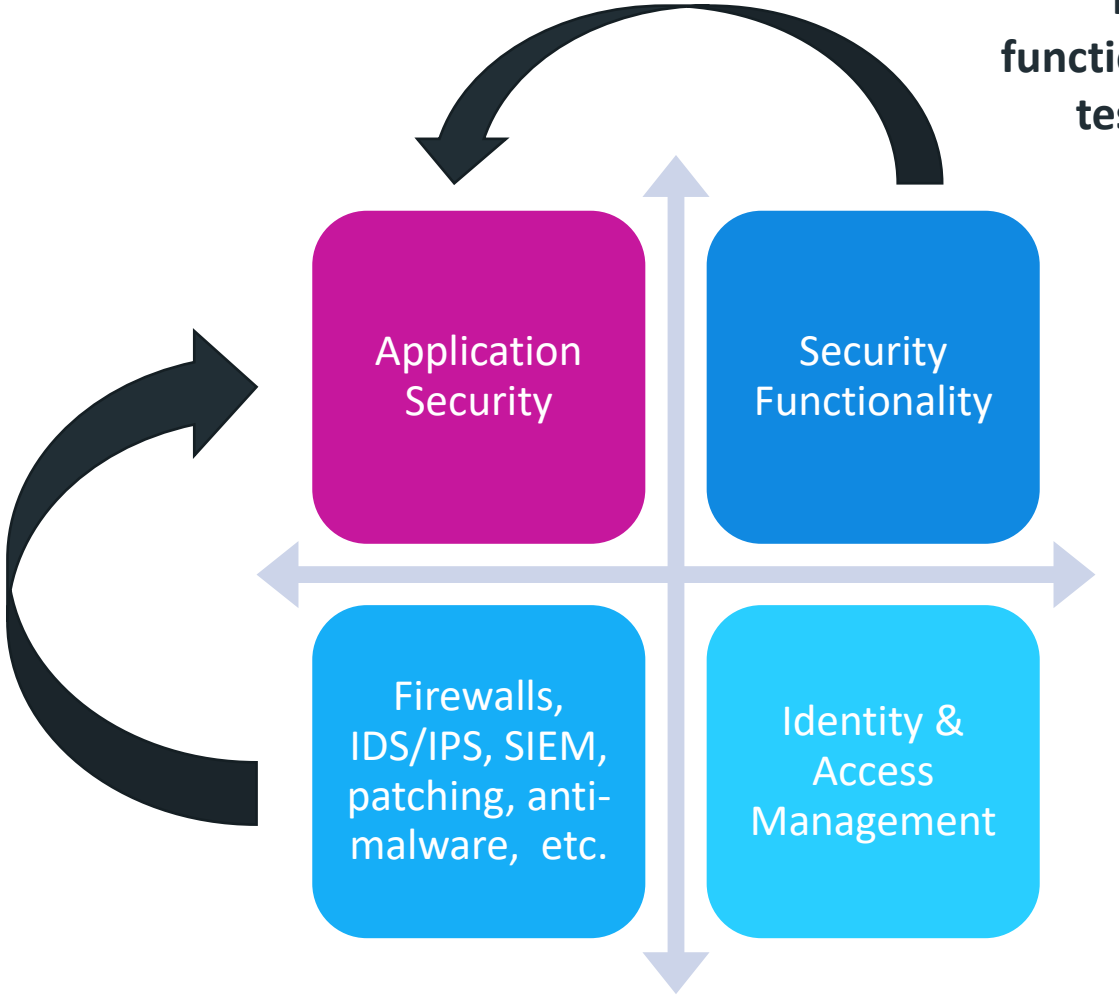


OWASP Top-10 2017

Injection	Broken Authentication	Sensitive Data Exposure	XML External Entities
Broken Access Control	Security Misconfiguration	Cross-Site Scripting	Insecure Deserialization
	Using Components with Known Vulnerabilities	Insufficient Logging & Monitoring	

AppSec needs specific attention

Testing for security functionality is different from testing for application security!

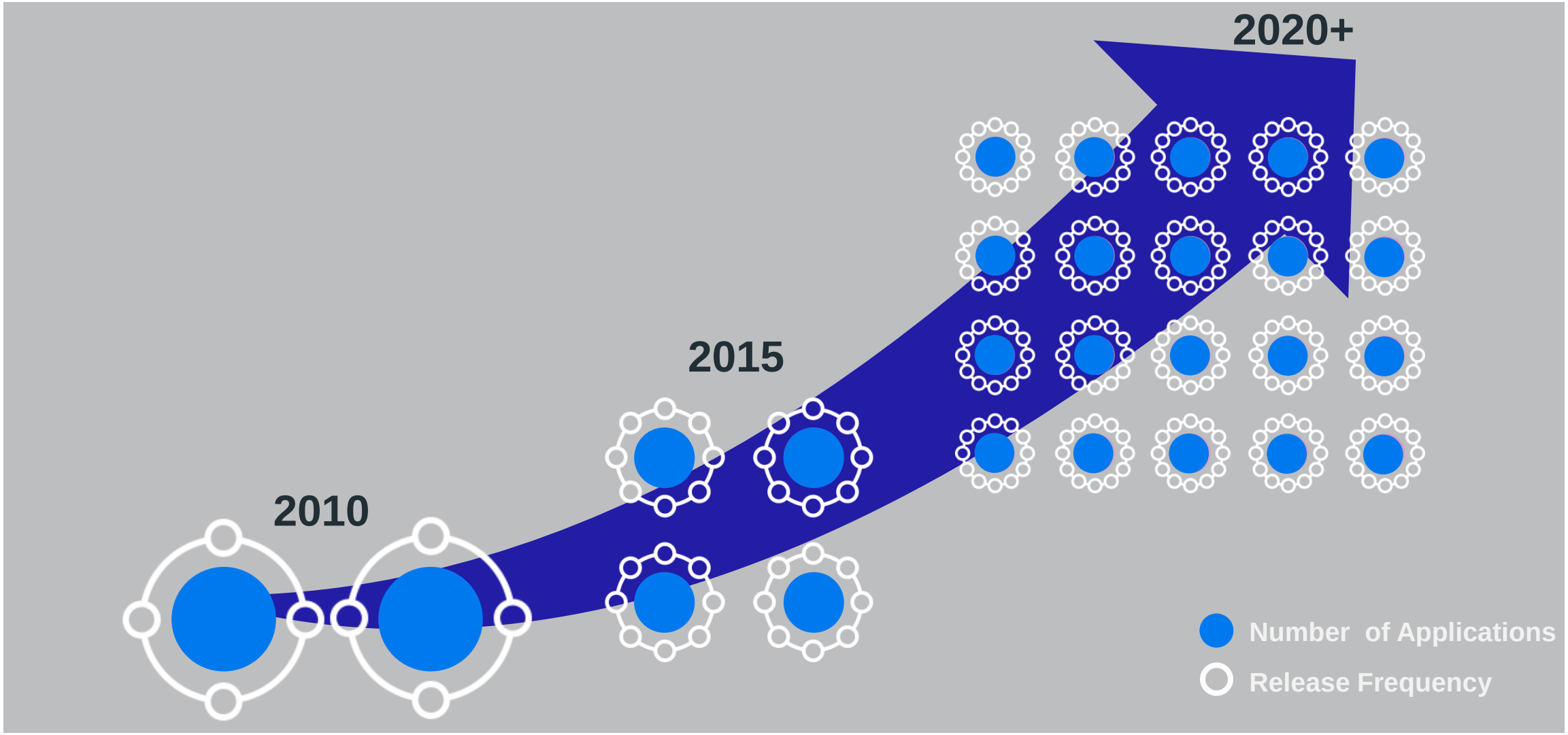


Infra-level security measures do not protect against this type of problem!

Factors making AppSec a big current issue

- Historically, most security investments have gone into infra. Remaining weak spots are in applications.
- Growing application portfolios and application connectivity.
- Lack of developer training and awareness.
- Rapid release cycles.

Manual pentesting and code reviews don’t offer needed scale and are too slow

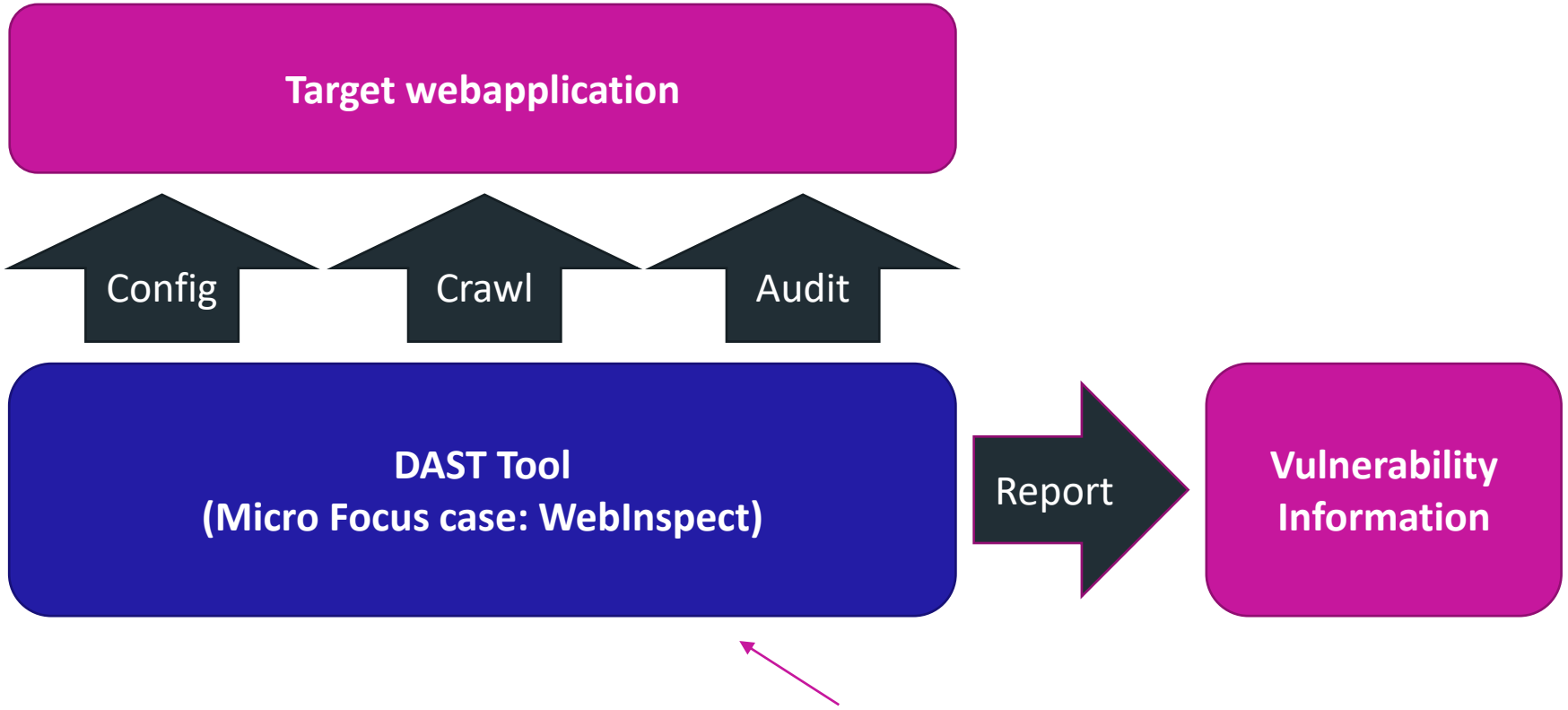


Testing for application security – techniques and challenges

Dynamic Application Security Testing (DAST)

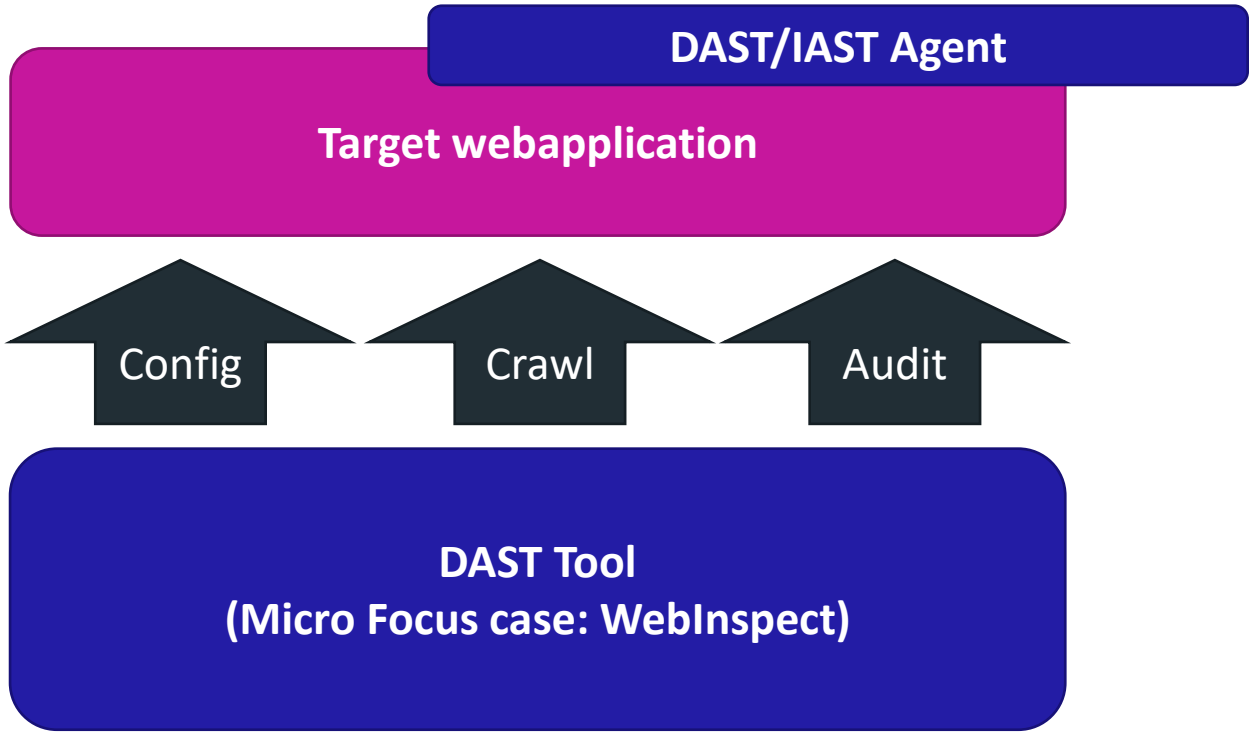
- Automatically testing a **running** application for security vulnerabilities.
- “Automated hacker”
- Usually done on test/QA environment, occassionally also done on production.

DAST process



Usually operated by security tester; sometimes run automatically from cmd line or API

IAST: Interactive Application Security Testing



"A helper behind enemy lines". Provides detailed info to the DAST tool to optimize its attacks.

DAST pros and cons

Pros

- Independent of programming language.
- In a way, similar to functional testing.
- Few “false positives”
- Can be done both manually and automated as part of a build pipeline.
- Can be integrated with functional testing tools and issue trackers.

Cons

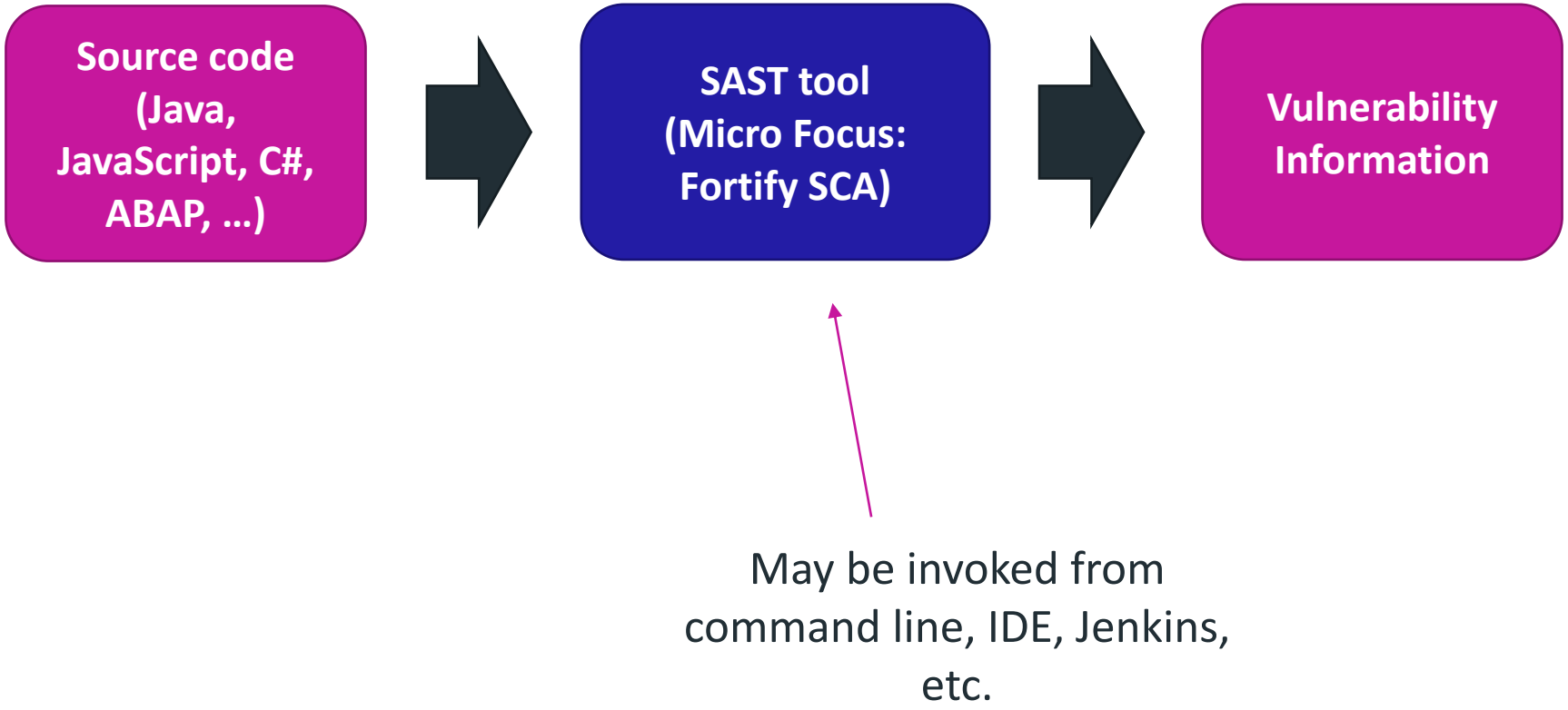
- Still relatively slow (several hours to days) and late in the cycle.
- Feedback in terms of behaviour – not super actionable for developers.
- Limited to web-based (HTTP) systems
- Needs to have the application running.
- Sensitive to configuration (log-in scripts, avoiding being hit by security controls).
- Prone to “false negatives” if configuration not correct.

Shifting left application security

Static Application Security Testing (SAST)

- Automatically analyzing the source code of an application for security vulnerabilities.
- “Automated code reviewer”
- Done based on code in the code repository; usually running automated every night.

SAST process



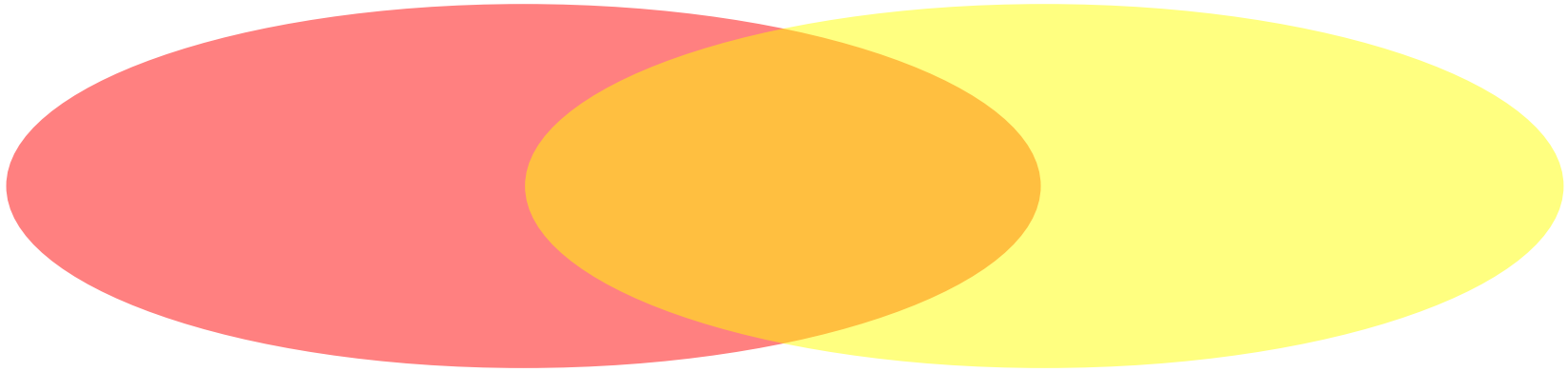
SAST versus static analysis for quality: Complementary solutions

SAST

- Fortify, Checkmarx, Veracode, Coverity, …
- Test for security, not for general quality.
- Slow, complex flow-analysis algorithms *plus* pattern-matching algorithms.

Static Analysis for Quality

- SonarQube, FxCop, CheckStyle, …
- Check for quality, with a bit of security.
- Fast, simple pattern-matching algorithms.



SAST pros and cons

Pros

- Fast (minutes to hours in extreme cases)
- Very detailed feedback to developers, easy to address issues.
- Web, mobile, desktop, embedded, ….
- Can find things that DAST cannot find.

Cons

- Prone to false positives.
- Requires that the programming language is supported by the SAST tool.
- Requires that the programming framework is understood by the SAST tool.
- Misses certain things that DAST can find.
- Fast, but still not real time.
- Not a good solution for 3rd party dependencies.

Two modern SAST developments

Software Composition Analysis (SCA)

- For most business apps, the custom code is just the tip of the iceberg: the majority of code is open source libraries!
- SCA is about testing the versions of the libraries against known vulnerable versions, and recommending patching.

- Micro Focus: integration with Sonatype, Snyk and others.

Real-time feedback

- Full SAST can't be done in real-time.
- Part of the SAST scanning *can* be done in real-time, providing immediate feedback to the dev inside the IDE.

- Micro Focus: Security Assistant

Combine SAST, DAST and sound production monitoring for complete application security.


MICRO FOCUS
FORTIFY
Thank you!