MICRO FOCUS FORTIFY

Automating application security with Fortify

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

FORTIFY®

Agenda

- Introduction to Application Security – what is and why care?
- Core appsec techniques: DAST and SAST
- Fortify products and implementation examples

Want to learn more after this?

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

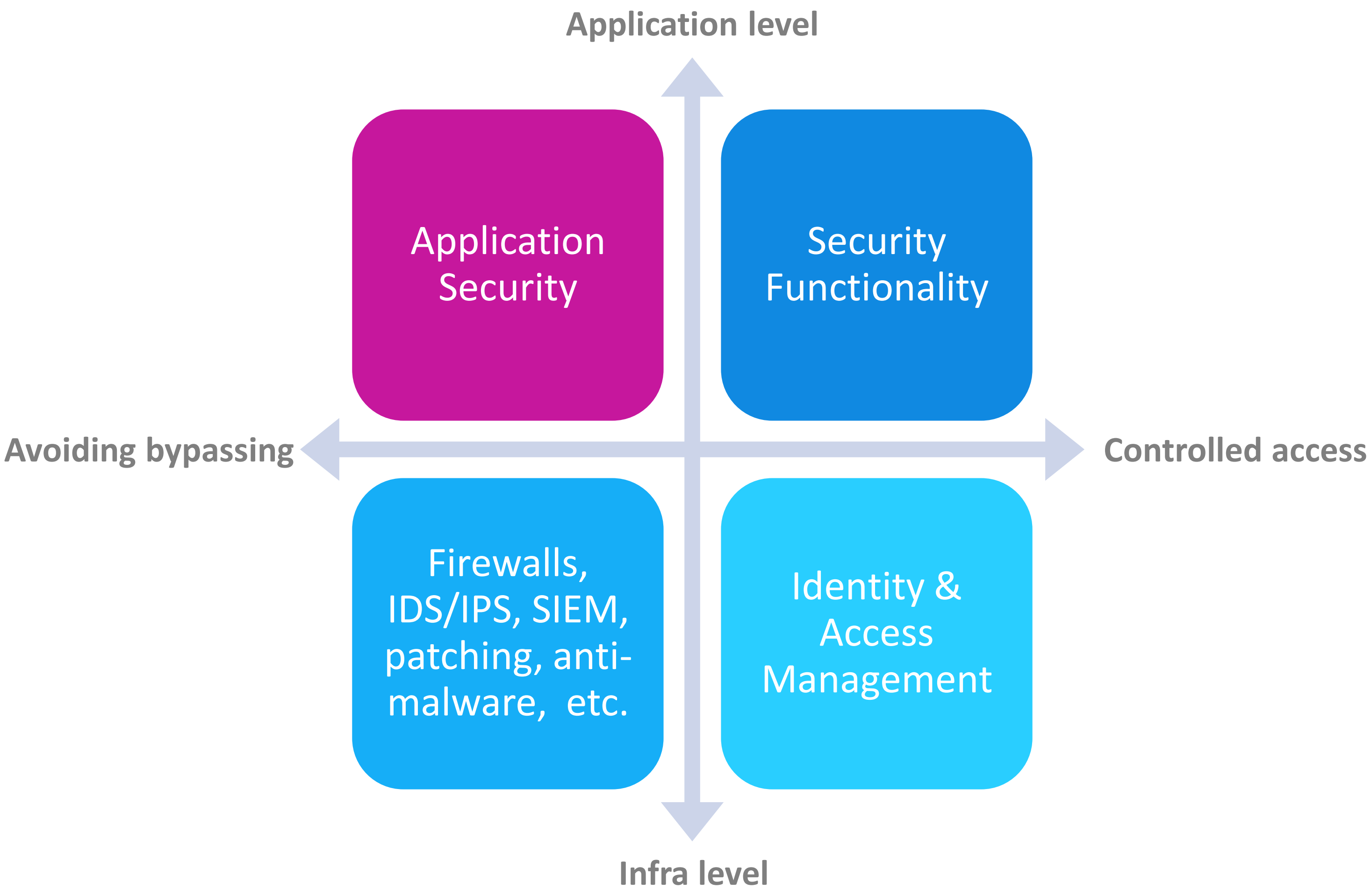
Introduction to Application Security – what is and why care?

The security of not bypassing security functionality.

Required for

The security of controlled access

A security quadrant

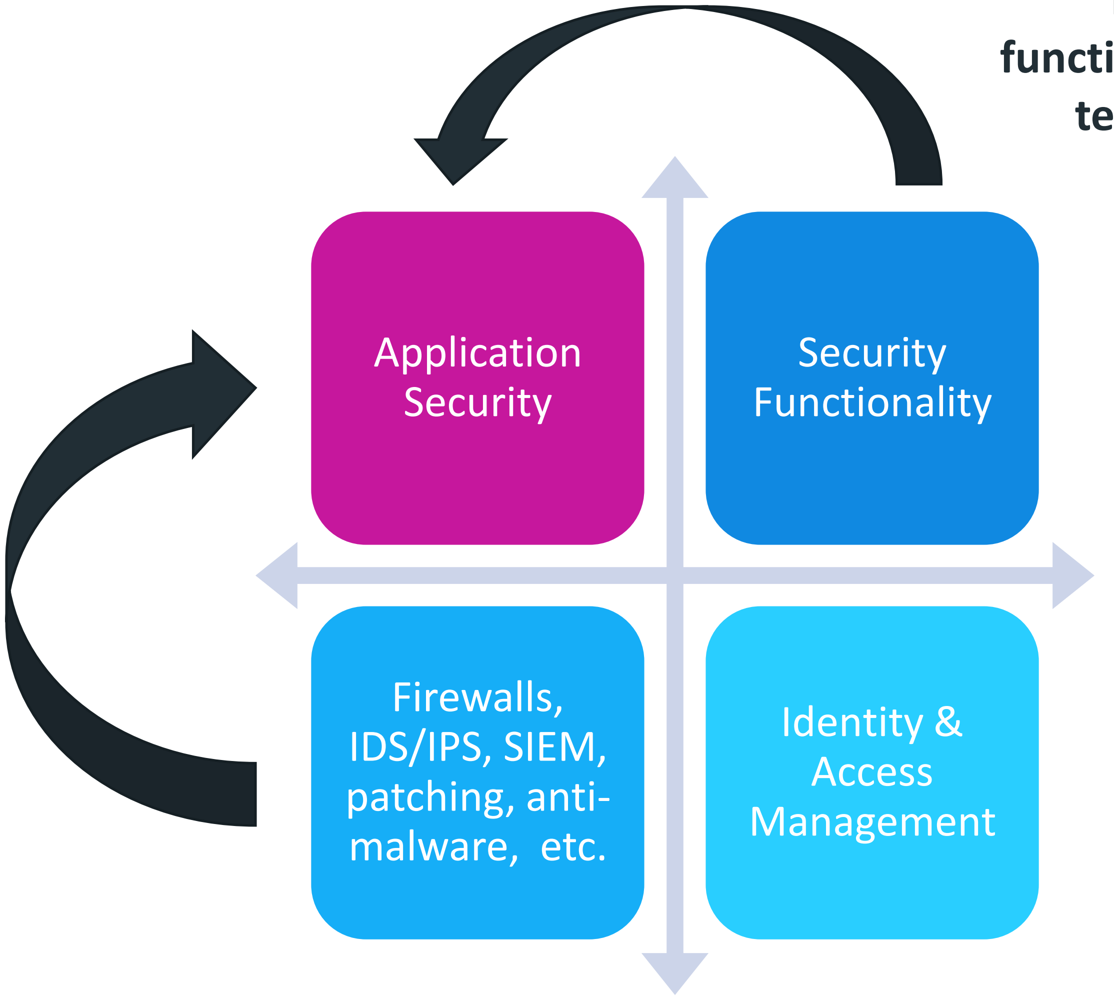


OWASP Top-10 2017

- 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!

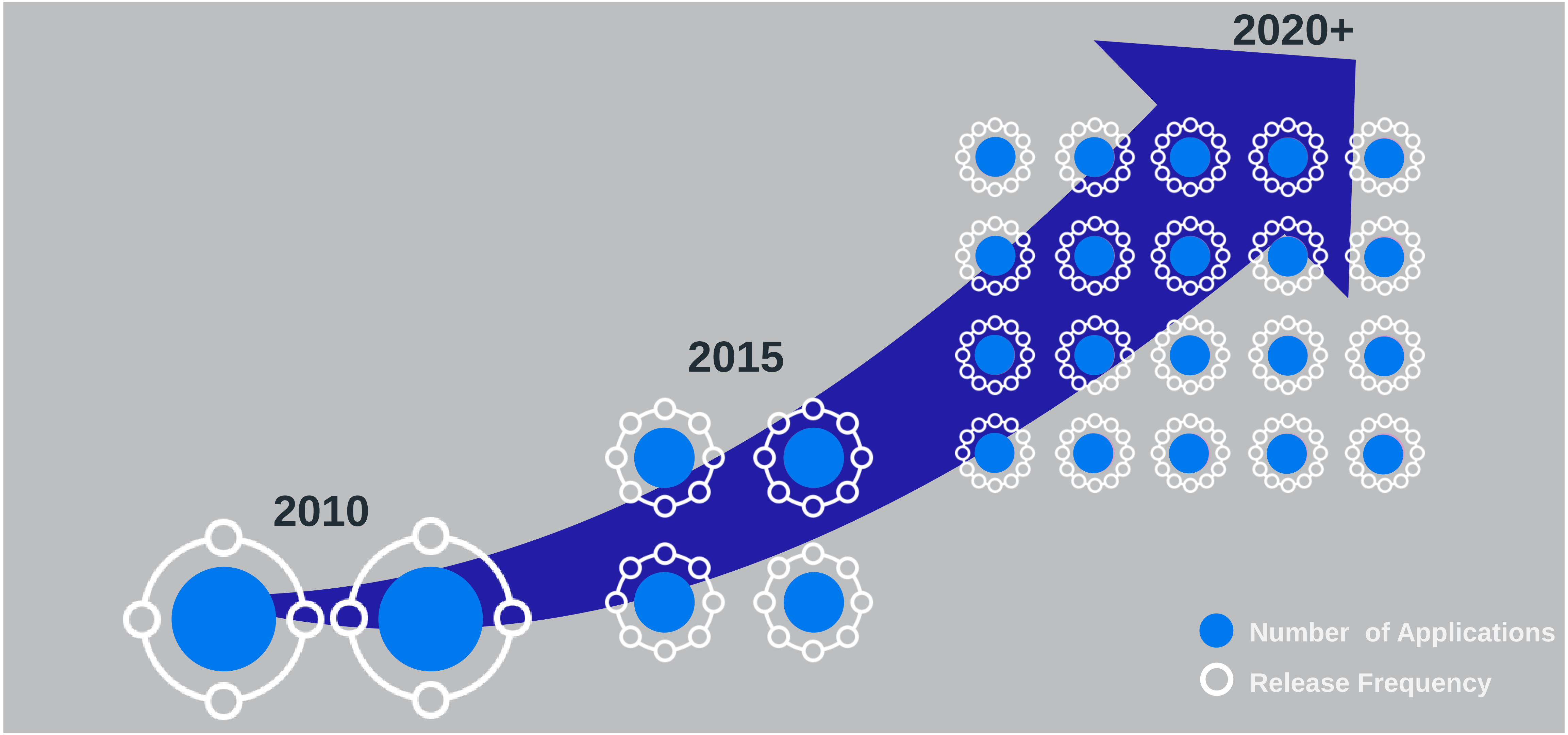

Application Security
Security Functionality
Firewalls, IDS/IPS, SIEM, patching, anti-malware, etc.
Identity & Access Management

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

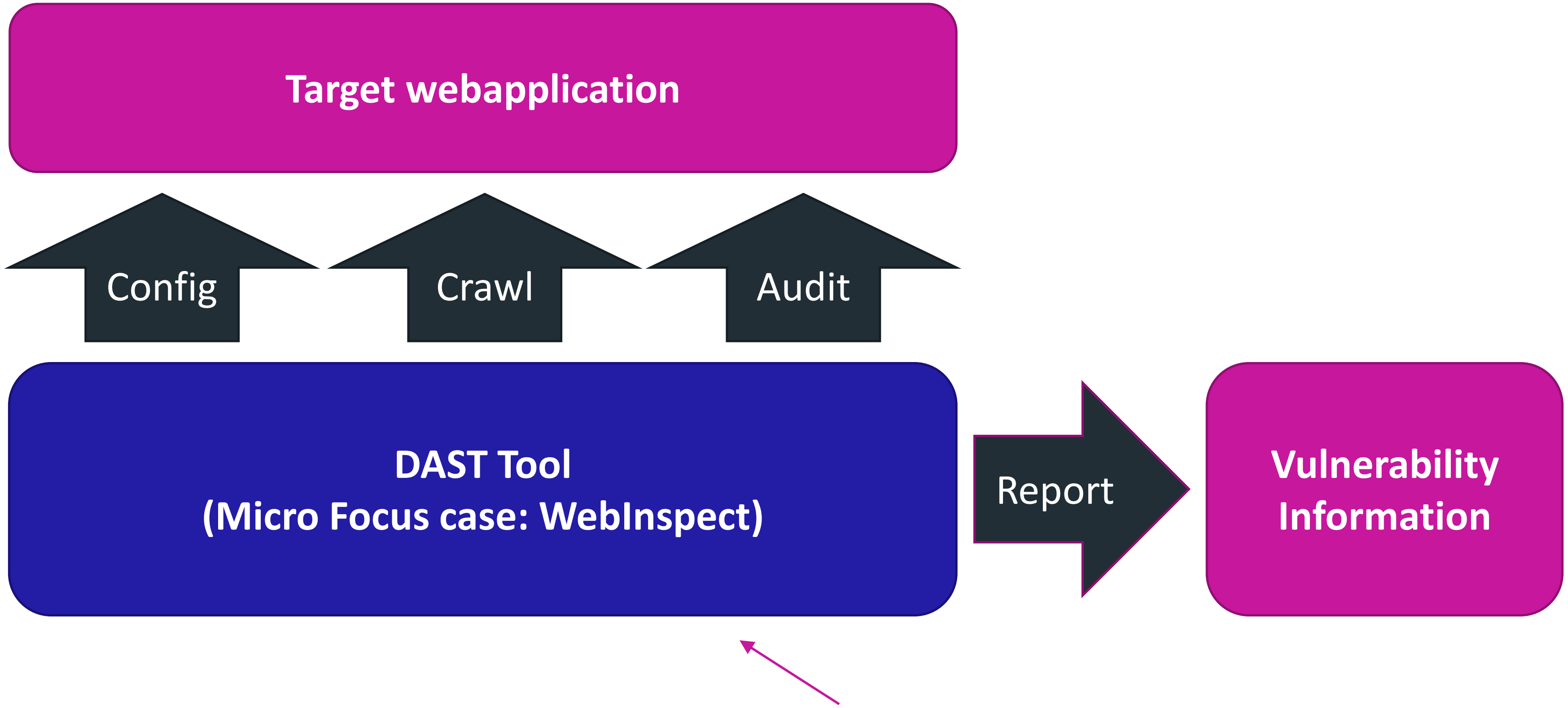


Core appsec techniques: DAST and SAST

Dynamic Application Security Testing (DAST)

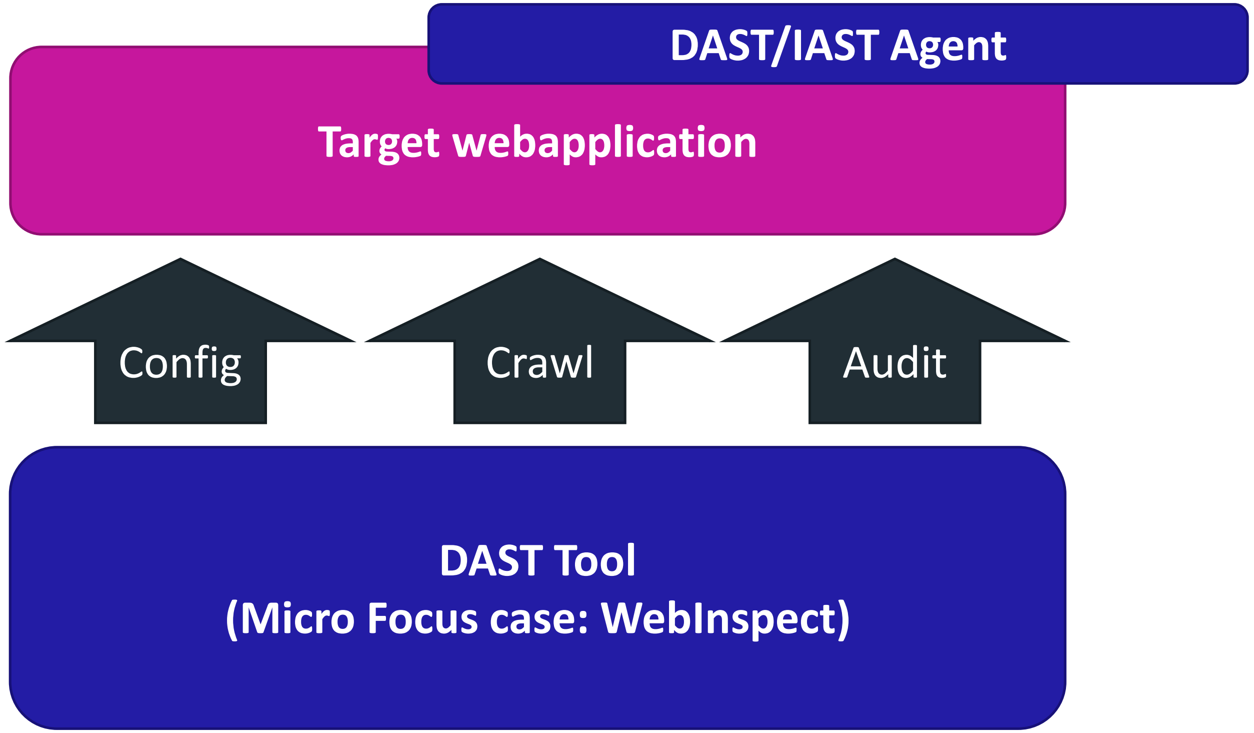
- Automatically testing a **running** application for security vulnerabilities.
- “Automated hacker”
- Usually done on test/QA environment, occassionally also done on production.

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.

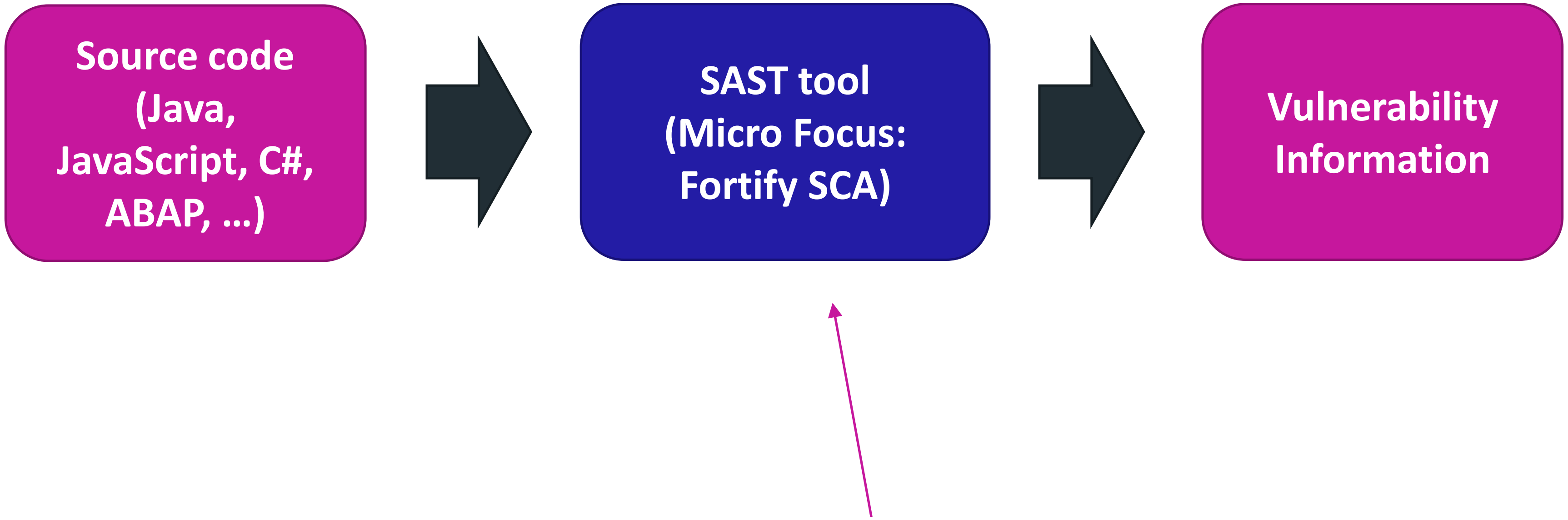
Cons

- 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.

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



Source code (Java, JavaScript, C#, ABAP, …)

SAST tool (Micro Focus: Fortify SCA)

Vulnerability Information

May be invoked from command line, IDE, Jenkins, etc.

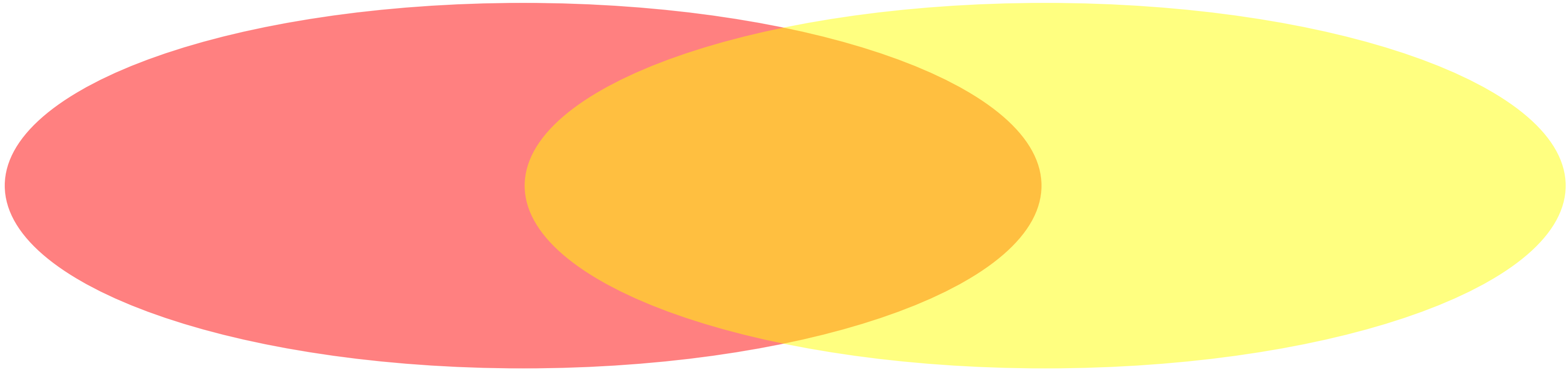
SAST versus static analysis for quality: Complementary solutions

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)

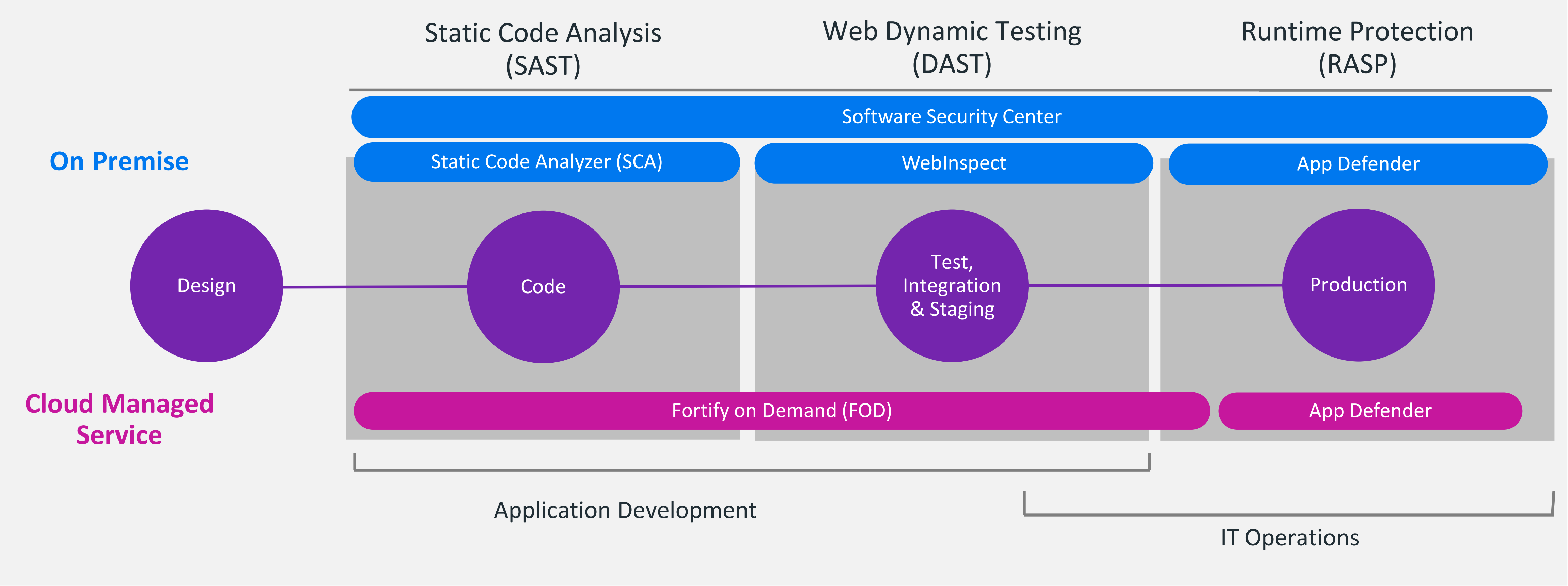
- For most business apps, the custom code is just the tip of the iceberg: the majority of code is open source libraries!
- SCA is about testing the versions of the libraries against known vulnerable versions, and recommending patching.
- Micro Focus: integration with Sonatype, Snyk and others.

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

Fortify products and implementation examples

Fortify is the most flexible, end-to-end AppSec solution



Fortify = Seamless Application Security

Easy to Get Started

- Start in a day with Fortify on Demand with actionable results

Easy to Use

- Real-time security in the IDE for developers with Security Assistant
- Robust integration ecosystem

Fast

- Get scan results in minutes
- Adjust scans to achieve desired coverage for both SAST and DAST
- Apply machine learning to identify and prioritize the most relevant issues with Audit Assistant

Accurate

- OWASP Benchmark: Fortify SCA true positive rate is 100%

Scalable

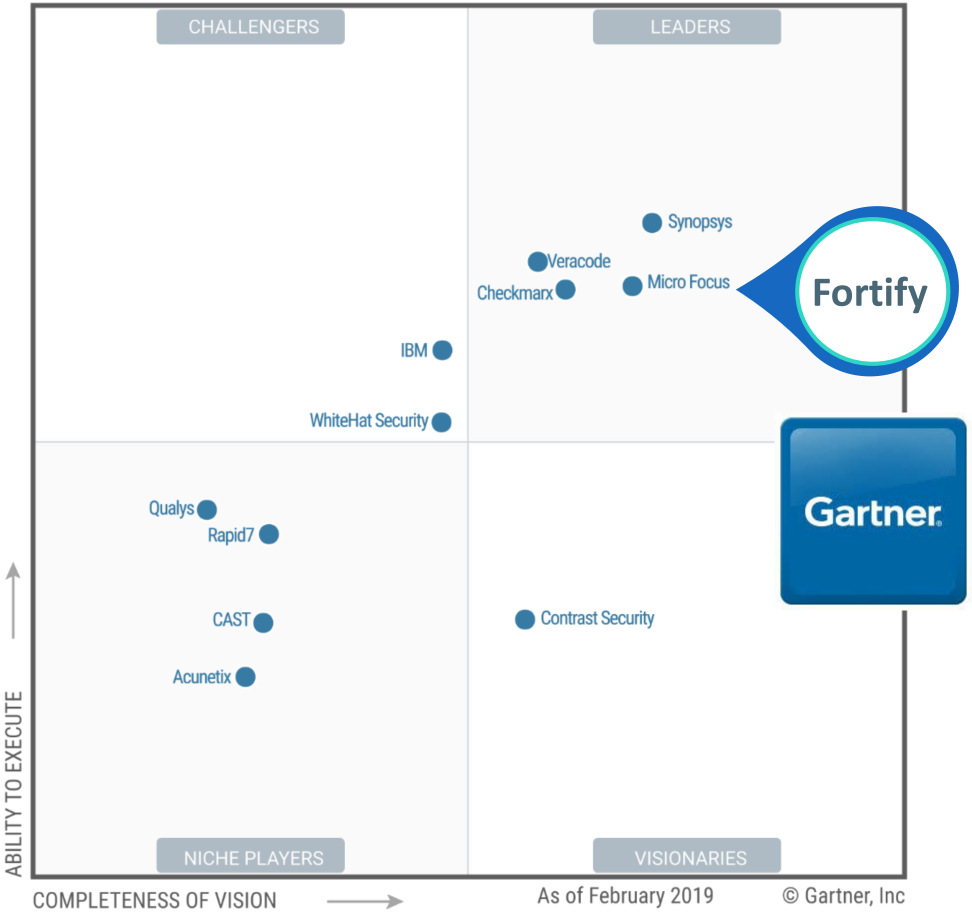
- SaaS, on-premise, or hybrid
- Flexible to grow

Fortify is recognized for delivering value

- Leader in Gartner MQ, and has been a leader in all editions of this MQ since they started it.
- Thousands of customers globally.
- Strong in financial services, independent software vendors, public sector, energy, automotive, telecommunications, consumer goods, and many other industries.

SAP acxiom ServiceMaster FICO
Aaron's novagalicia banco Affinity Credit Union nielsen
Cox Automotive DELTA Heartland Payment Systems centrica British Gas

2019 Gartner Magic Quadrant for AST



As of February 2019 © Gartner, Inc

Example scenario 1: Small supplier to healthcare industry

Imagine the following prospect

- 100 employees
- Sells technical equipment to healthcare industry customers
- Has 4 web applications (public website, support portal, …)
- IT manager is conscious about security, because their customers are as well. Hires external agency for pentesting once a year.

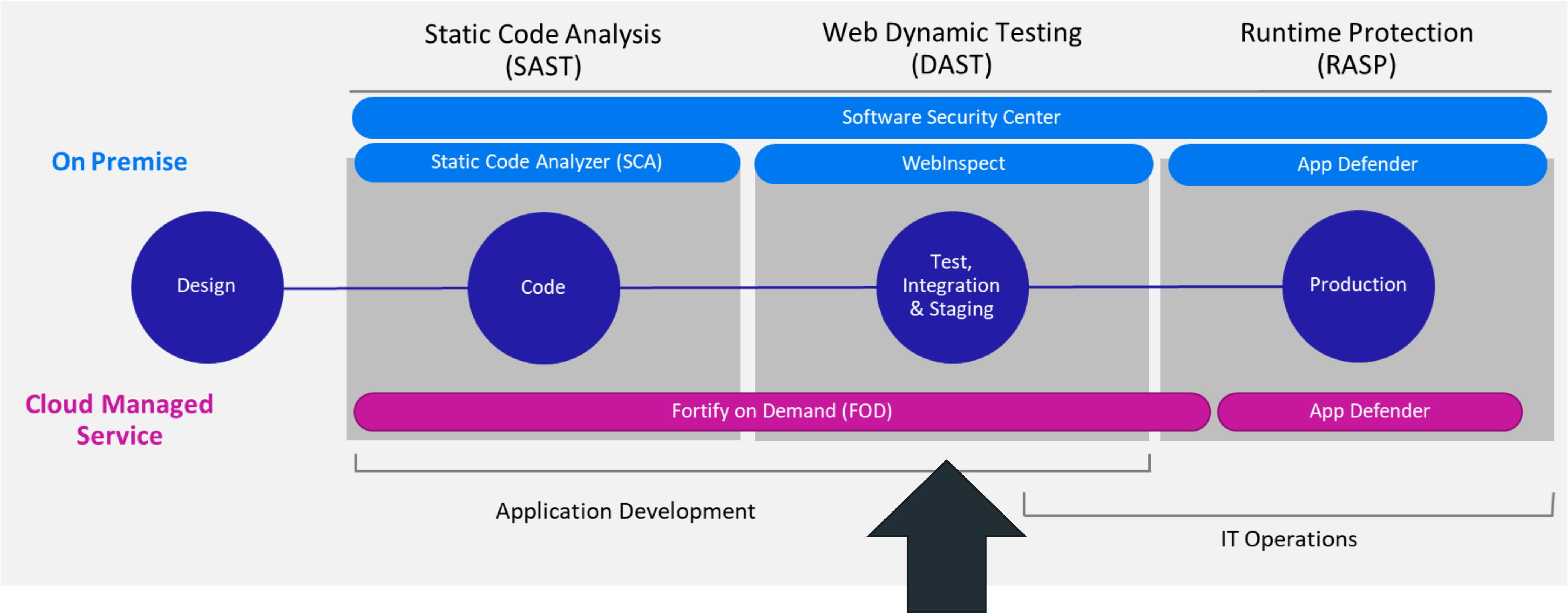
Typical pain points

Scenario 1: Small supplier to healthcare industry

- Penetration testing is **expensive**.
- Quality of penetration testing report is highly variable.
- They really want to do it **more often**, but at the present cost level this is not feasible.
- They expect to launch 2 more applications next year, so getting a practical, scalable solution is important.

Fortify solution: FoD dynamic

Scenario 1: Small supplier to healthcare industry



Example scenario 2: A bank introducing DevOps

Imagine the following prospect

- Bank with 5.000 employees, of which 400 software developers
- Maintain 50 applications (web, mobile apps, internal systems, etc.)
- Have an application security department.
  - Regularly perform code reviews
  - Run dynamic testing tools themselves and hire 3rd party experts for additional testing.
- Currently in the process of introducing DevOps for quicker time-to market.

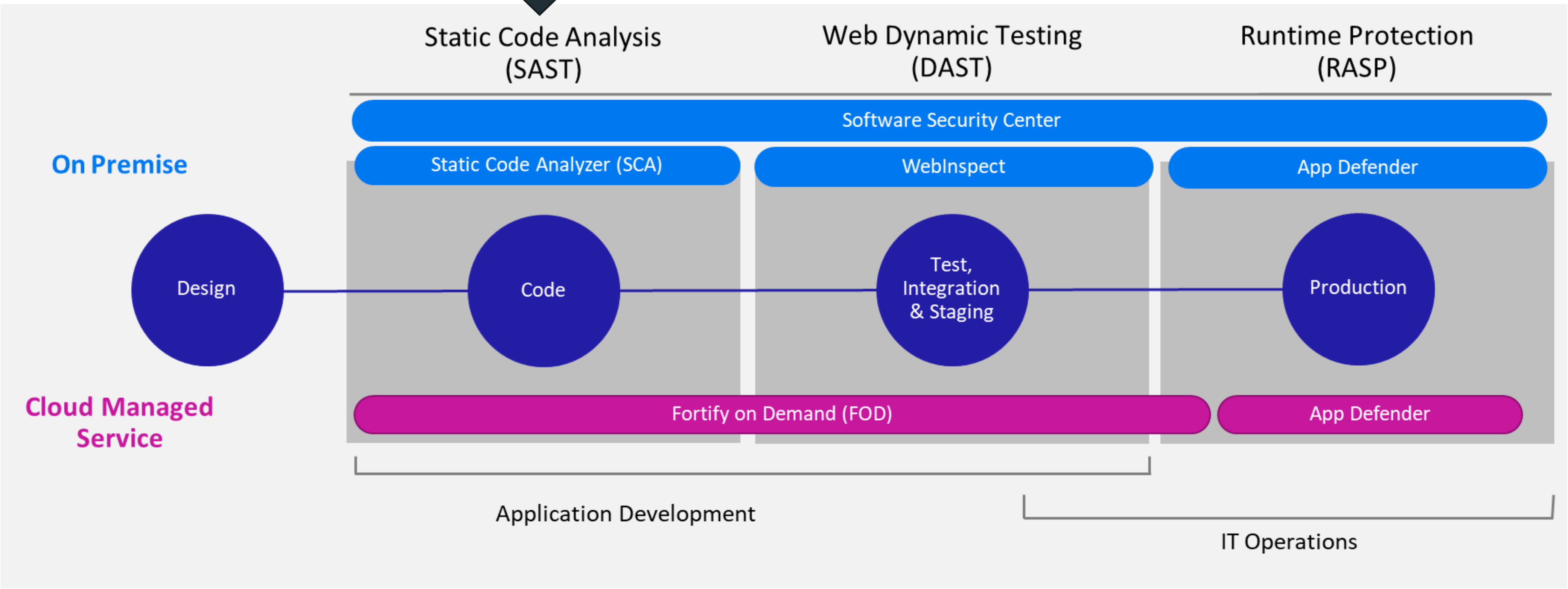
Typical pain points

Scenario 2: A Bank introducing DevOps

- The current security process will become the bottleneck in the DevOps process. Something needs to be done.
- Regulatory pressure to maintain a high level of security.
- Developers are under a lot of pressure to deliver functionality for the business. They dislike the security processes.
- Code review is important, but at the same time the code is a strategic asset not to be shared with 3rd parties.

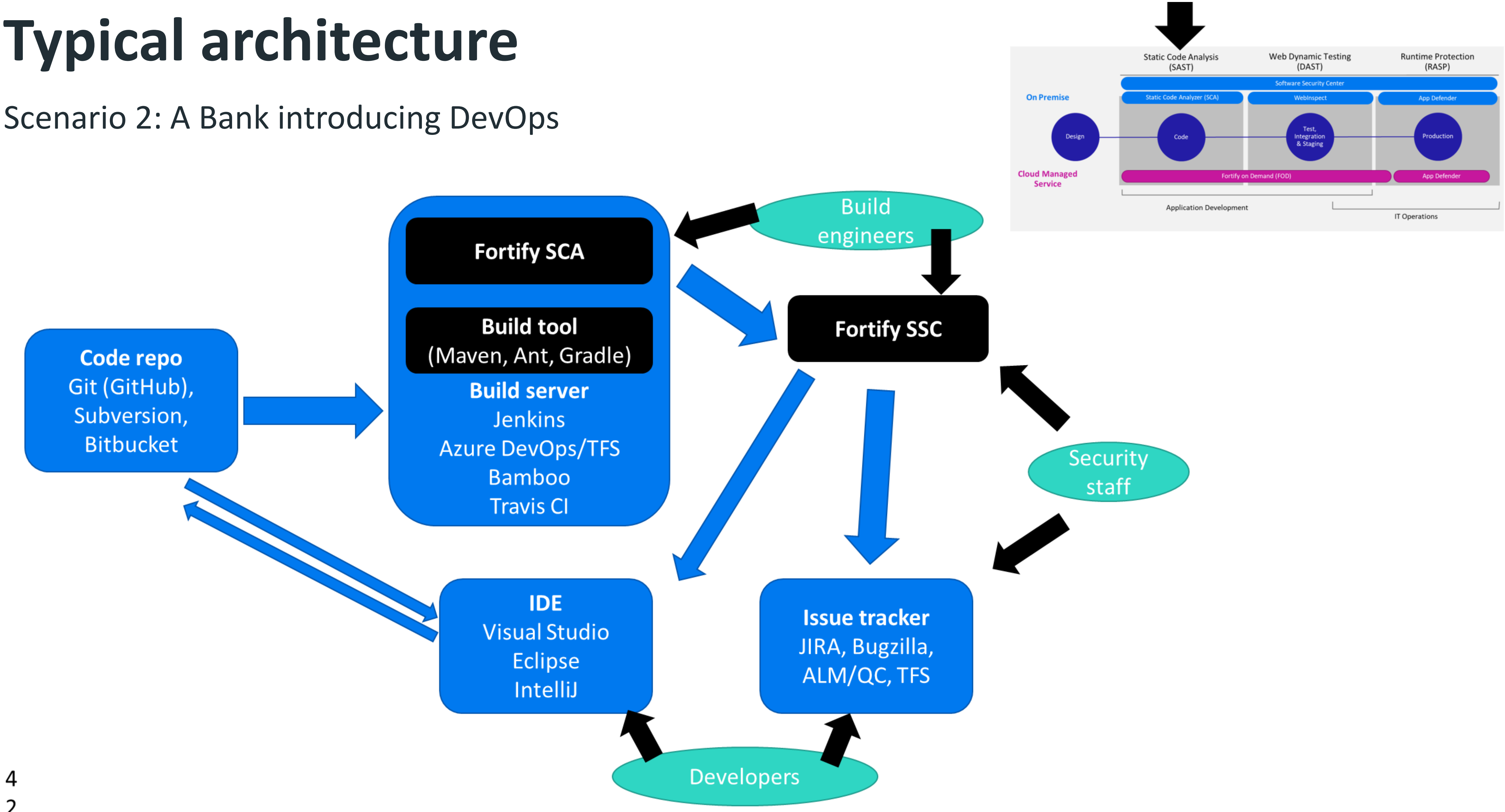
Fortify solution: SAST on-premise

Scenario 2: A Bank introducing D[illegible]ps



Typical architecture

Scenario 2: A Bank introducing DevOps



4
2

Conclusion

- Application Security as a topic cannot be ignored by organizations that operate custom software.
- Manual approaches to the problem exist but are painful in terms of cost, scalability and the delays they introduce.
- Fortify is Micro Focus’ market-leading appsec automation portfolio.
- With SAST/DAST/RASP available on-prem and as-a-service, there’s an effective solution for any type of situation.

MICRO FOCUS® FORTIFY

Thank you!