Managed Application Security

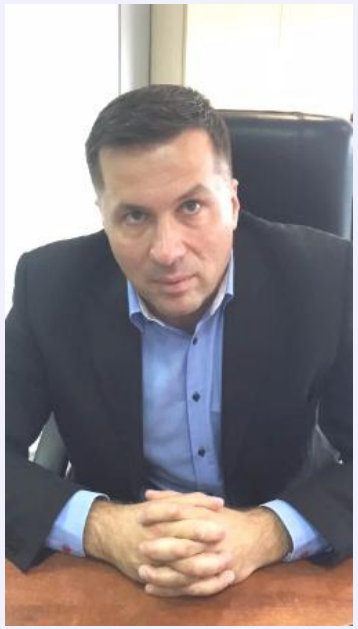
trends and best practices in application security

Adrian Locusteanu, B2B Delivery Director, Telekom Romania
adrian.locusteanu@telekom.ro

OWASP
The Open Web Application Security Project

About Me

OWASP
The Open Web Application Security Project

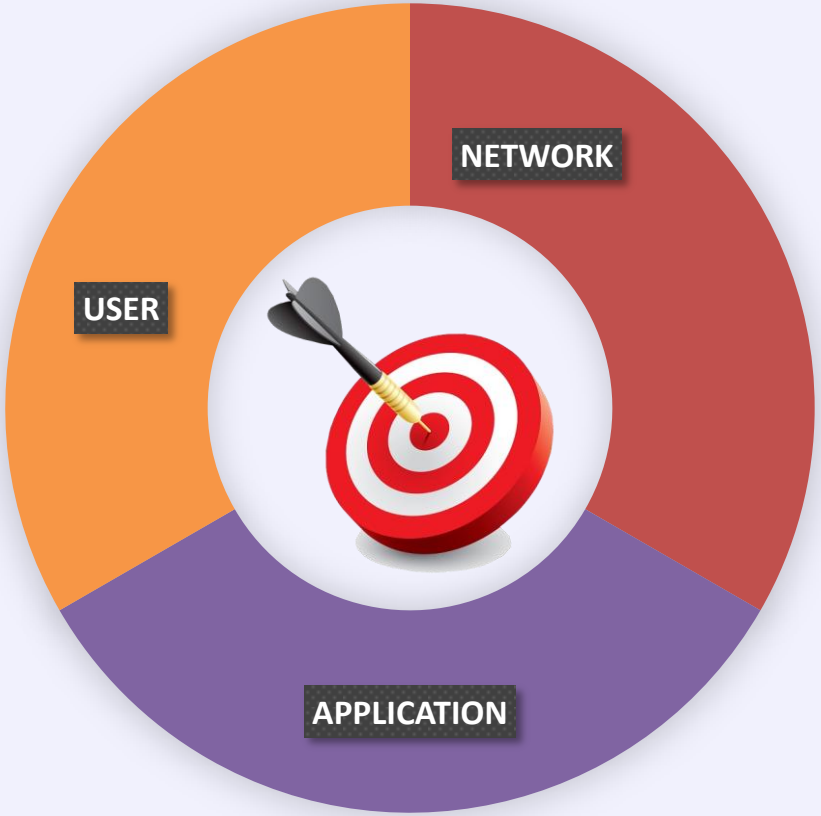


Adrian Locusteanu is the B2B Delivery Director of Telekom Romania.

His background includes the management of selling & delivery of ICT projects within multi-cultural enterprise environment having more than 20 years of experience in the ICT solutions and services market for government & Top100 enterprises.

Adrian graduated Facultatea Automatica (UPB) and Academia de Studii Economice. He also holds an Executive MBA degree, a Master in Information Security and is a member of the Association of Chartered Certified Accountants.

MAIN SECURITY ATTACK VECTORS



OWASP
The Open Web Application Security Project

COMPELLING & BASIC TRUTHS ABOUT APPLICATION SECURITY

- **Top breaches*:**
  - **Web Application Attacks 30%**
  - **CyberEspionage 14.93%**
  - **Privilege Misuse 14.3 %**
  - **Miscellaneous Errors 11.5%**

- **Top incidents*:**
  - **Denial of Service 26.7%**
  - **Privilege Misuse 18.4%**
  - **Crimeware 16.5%**
  - **Web Application Attacks 11.5%**

** source: Verzione Data Breach Investigations Report 2017*

- Application Security represents the highest risk attack vector with the least amount of strategic planning and spend (read opportunity!!)

- Attack surface expands as all organizations are continuously increasing web presence and application spend in order to optimize business

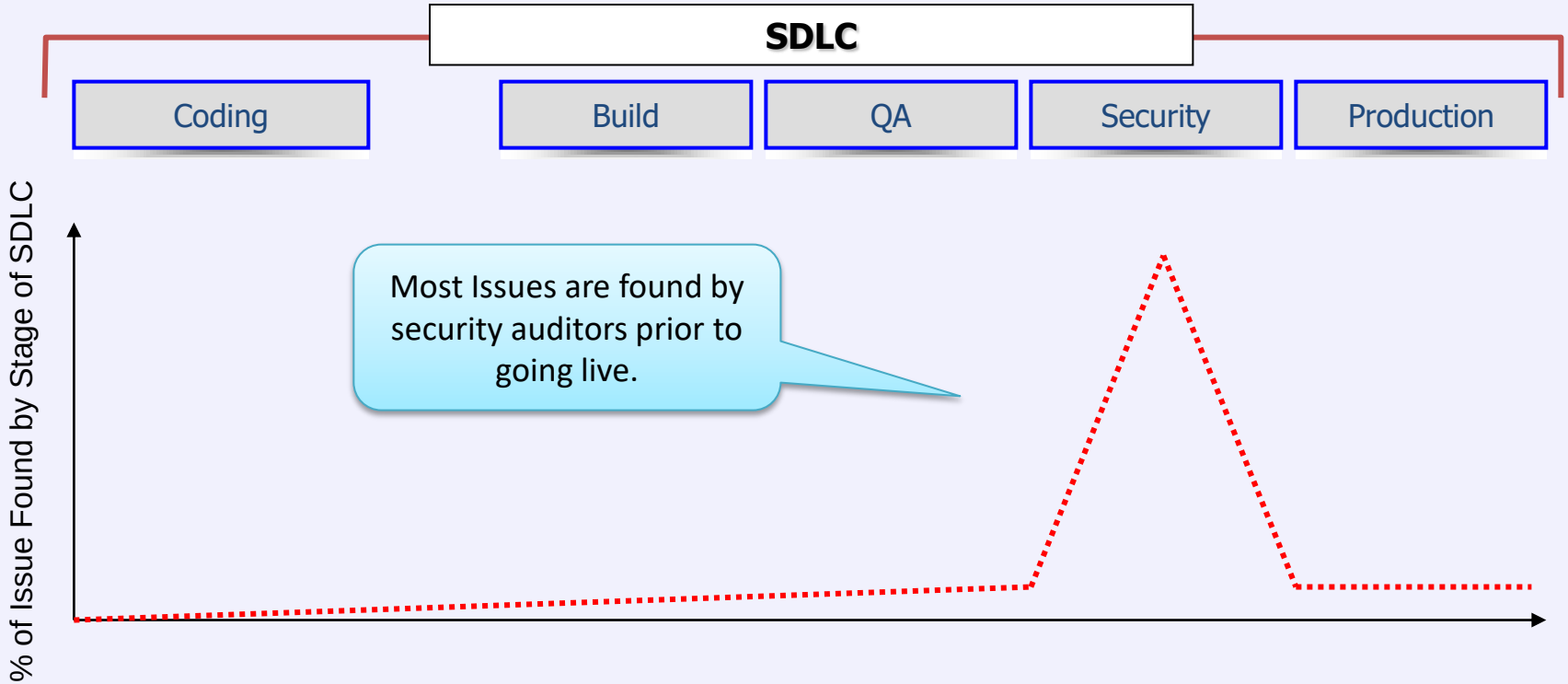
OWASP
The Open Web Application Security Project

OWASP Top Application Security Risks

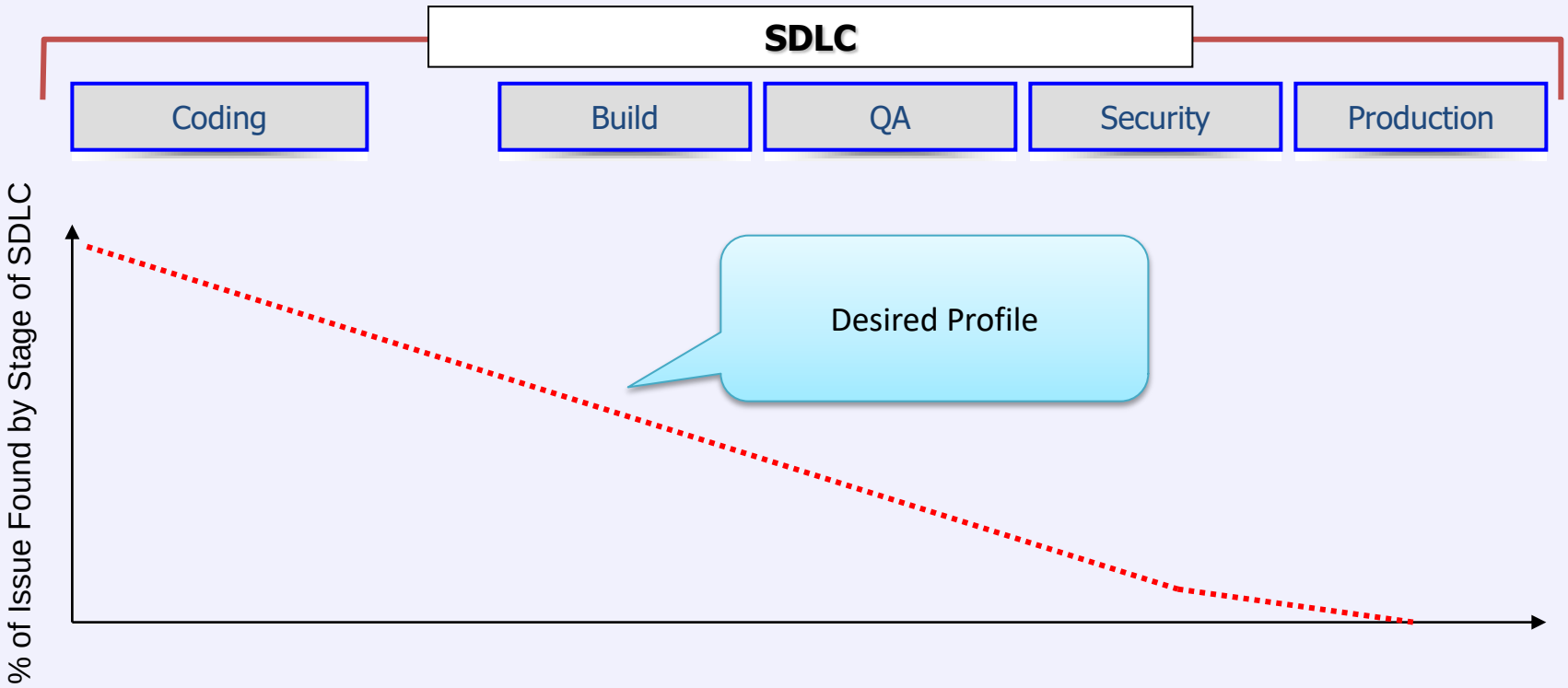
2013	2017
A1 Injection	A1 Injection
A2 Broken Authentication and Session Management	A2 Broken Authentication and Session Management
A3 Cross-Site Scripting (XSS)	A3 Cross-Site Scripting (XSS)
A4 Insecure Direct Object References	A4 Broken Access Control
A5 Security Misconfiguration	A5 Security Misconfiguration
A6 Sensitive Data Exposure	A6 Sensitive Data Exposure
A7 Missing Function Access Level Control	A7 Insufficient Data Protection
A8 Cross-Site Request Forgery	A8 Cross-Site Request Forgery
A9 Using Components with Known Vulnerabilities	A9 Using Components with Known Vulnerabilities
A10 Unvalidated Redirects and Forwards	A10 Underprotected APIs

- Top3 vulnerabilities remain unchanged
- Controversed new A7 (Insufficient Data Protection)
- A10 (Underprotected APIs) reflecting technology evolution (IoT, Cloud, etc…)

Security Testing Within the Software Lifecycle



Security Testing Within the Software Lifecycle



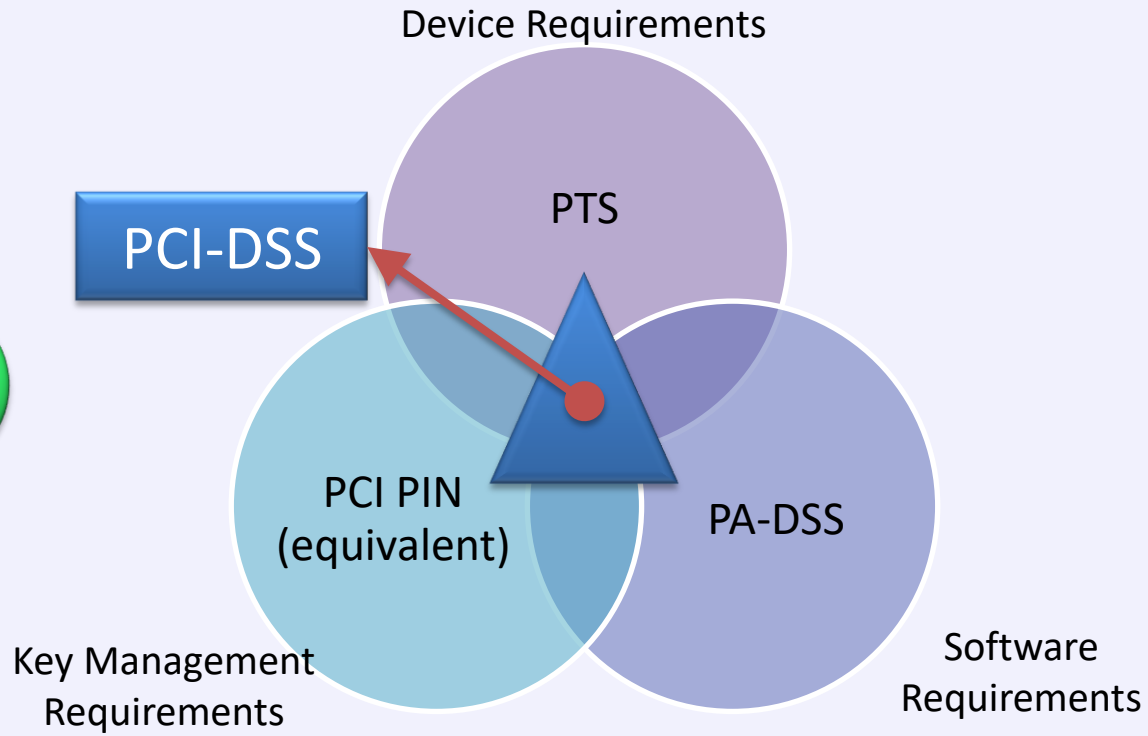
OWASP
The Open Web Application Security Project

TYPICAL DEVELOPMENT CYCLE SHORTCUTS and issues

- **Ambitious time-to-market puts pressure on security testing schedule**
- **Compromise on security to reach desired functionalities**
- **Deviations from security development methodologies**
- **No investment in specialized testing tools**
- **Not involving specialized security consultants in testing process**
- **Insufficient or no security training/awareness for developers**

HOW SHOULD APPLICATION SECURITY BE APPROACHED

an example from a related area

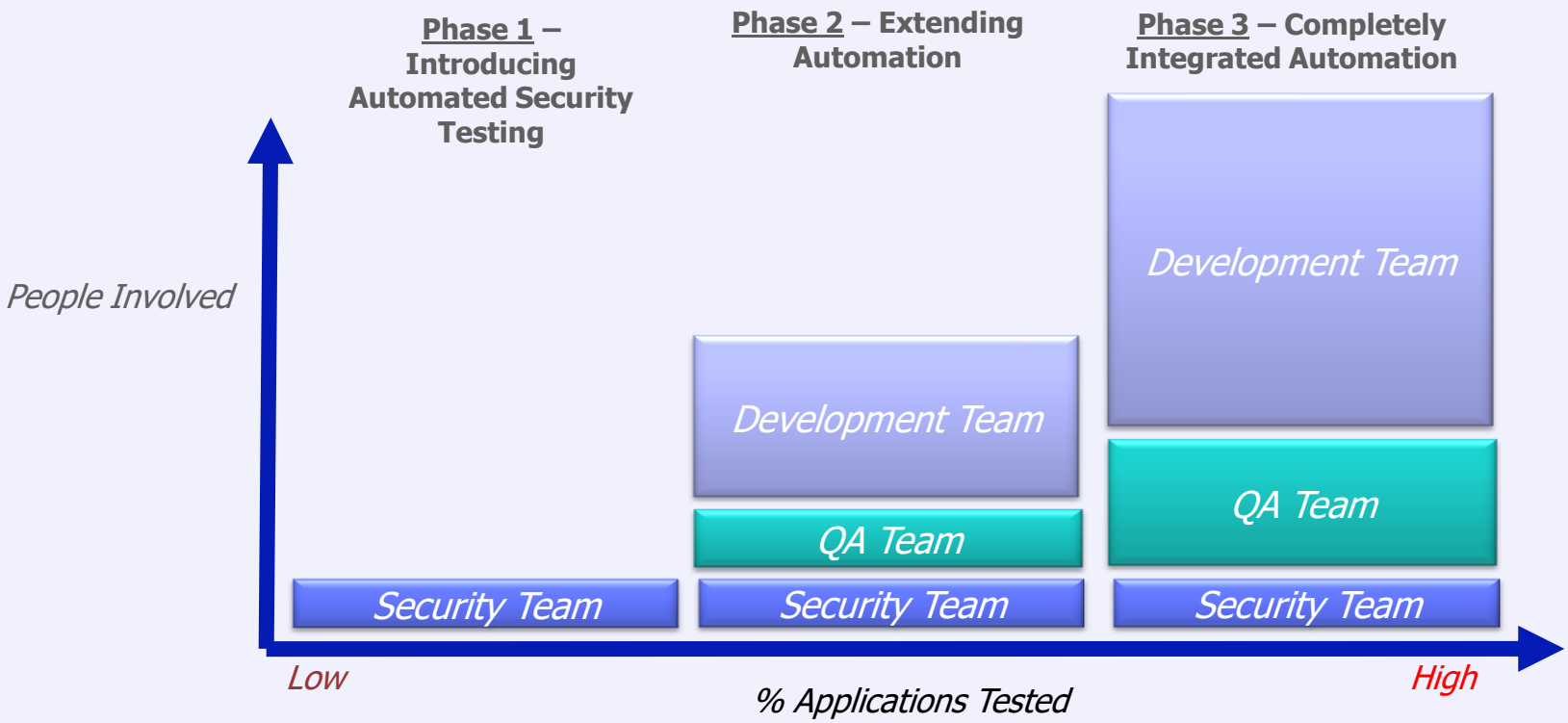


HOW SHOULD APPLICATION SECURITY BE APPROACHED

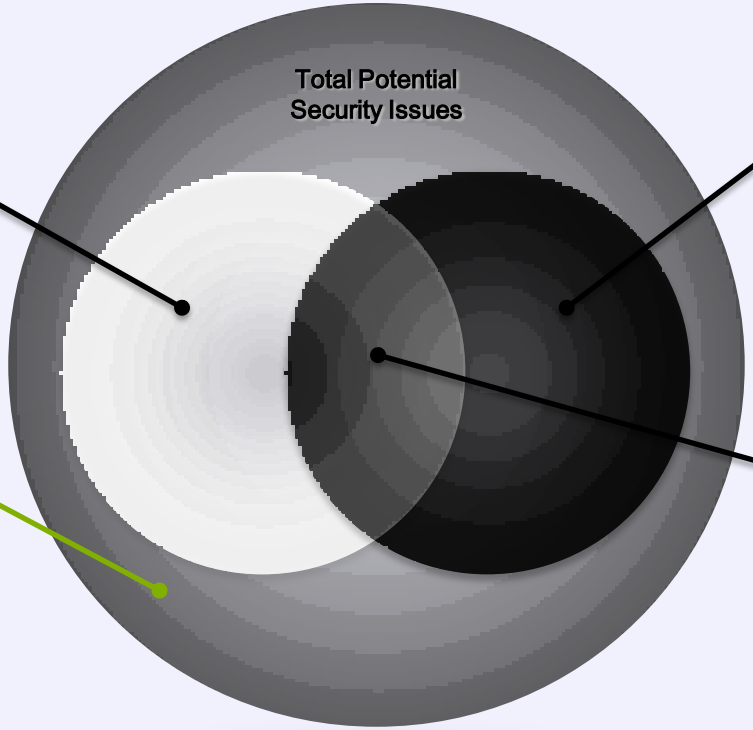
Lessons learned from POS Application

- **End2End secured environment** : strict and inter-related security requirements at all levels (hardware, kernel, key management, communication, software)
- **Standardized application security testing**: Visa/Mastercard application testing
- **Control mechanisms (audits), discipline and penalties**

The Need to Scale Security Testing



DAST and SAST – Issue Type Coverage



SAST Only

- Null pointer dereference
- Threading issues
- Code quality issues
- Issues in dead code
- Insecure crypto functions
- Issues in back-end application code
- Complex injection issues
- Issues in non-web app code

DAST Only

- Environment configuration issues
- Patch level issues
- Runtime privileges issues
- Authentication issues
- Protocol parser issues
- Session management issues
- Issues in 3rd party web components
- Cross-site request forgery
- Malware analysis

Manual Testing

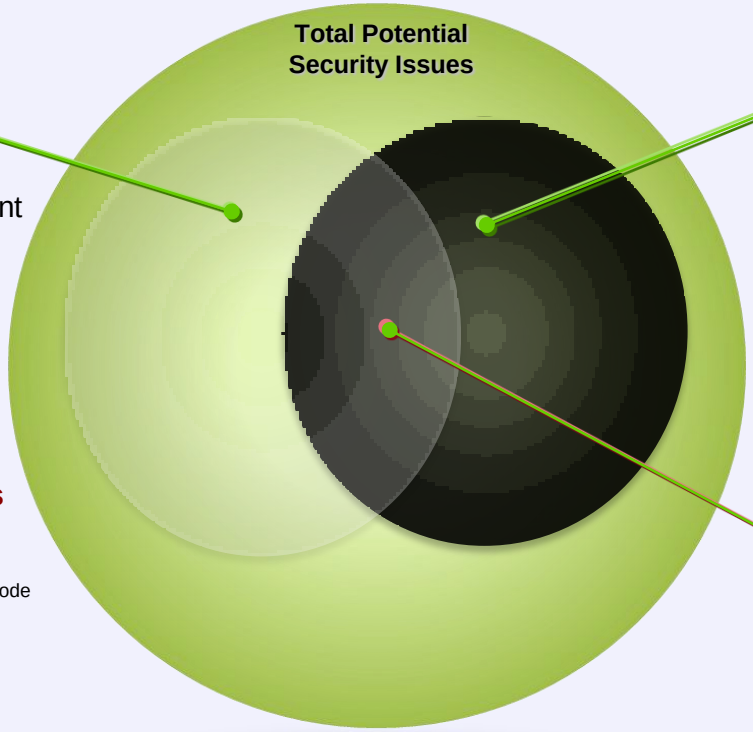
- Business logic issues

DAST & SAST

- SQL Injection
- Cross Site Scripting
- HTTP Response Splitting
- OS Commanding
- LDAP Injection
- XPath Injection
- Path Traversal
- Buffer Overflows
- Format String Issues
- ...

OWASP
The Open Web Application Security Project

Find more vulnerabilities using the most advanced techniques



Static Analysis

- Analyze Source Code
- Use during development
- Uses Taint Analysis / Pattern Matching

Dynamic Analysis

- Analyze Live Web Application
- Use during testing
- Uses HTTP tampering

More adv. Techniques

Client-Side Analysis

- Analyze downloaded Javascript code which runs in client
- Unique in the industry

Run-Time Analysis

- Combines Dynamic Analysis with run-time agent
- More results, better accuracy

Hybrid Analysis

- Correlate Dynamic and Static results
- Assists remediation by identification of line of code

OWASP
The Open Web Application Security Project

Advanced security testing collaboration & governance through application lifecycle

Blackbox security testing

End2end security testing

Whitebox code analysis

RECENT DRIVERS/Constraints FOR APPLICATION

GDPR

EU General Data Protection Regulation

SECURE TRANSPARENCY

The data subject needs to know what personal information we collect, we manipulate, to what purpose, and have control in the process.

All personal data should be secured and remain private during the entire lifecycle.

INFORMATION LIFECYCLE MANAGEMENT

policy-based approach to managing the flow of information through a life cycle from creation to final disposition.

GDPR was developed to ensure organization deal with personal information in a responsible manner

TRANSPARENCY

GDPR was developed to ensure the end user has visibility to his data

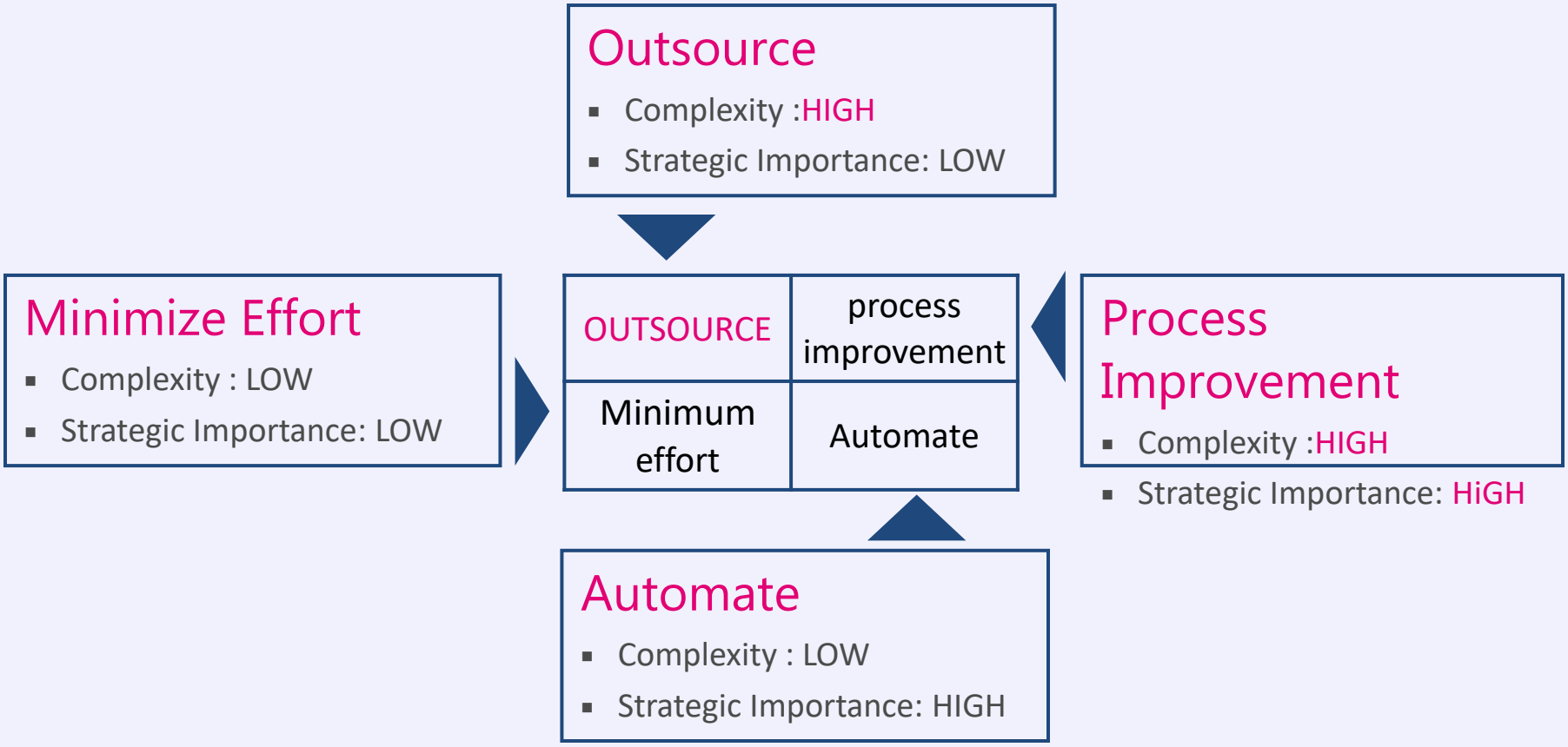
SECURITY

GDPR was developed to ensure the end user that his personal information remains private

SECURITY BY DESIGN !

SECURITY BY DEFAULT !

DECISION: IN-HOUSE VERSUS OUTSOURCE



SERVICE CENTRIC APPLICATION SECURITY