



OWASP

Open Web Application  
Security Project

# Agile yes, but secure?

OWASP Meeting Stuttgart

3.8.2015

# About me



Andreas Falk  
NovaTec Consulting GmbH  
[andreas.falk@novatec-gmbh.de](mailto:andreas.falk@novatec-gmbh.de)

 @NT\_AQE  
@Agile\_Security



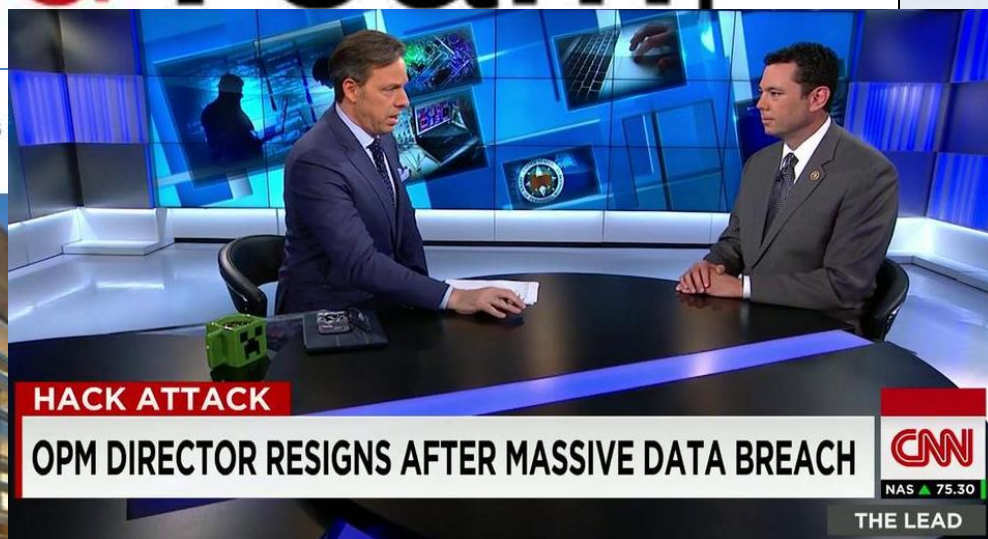
# Who's next to be hacked?

**Hacking Team** -- purveyor of exploits and spyware to a variety of **government agencies** all over the world -- has been hacked. Late Sunday night, its **Twitter account name** was changed to "Hacked Team" and its bio to read:

# ]HackedTeam[



**Help Net Security** @helpnetsecurity · 3 Std.  
Hackers hit UCLA Health, access medical files of 4.5 million patients  
[bit.ly/1VI07CO](http://bit.ly/1VI07CO) - @Lancope @OPSWAT



8/3/2015

Chapter Meeting Stuttgart



# Who's next to be hacked?

## 19 Online Cheating Site AshleyMadison Hacked

JUL 15



Large caches of data stolen from online cheating site Ashley Madison were posted online by an individual or group that stole data from the company's user databases, financial records and other information. The unfolding leak could be quite damaging to the company, whose slogan is "Life is short. Have an affair."



**Adult FriendFinder**

@adultfriendfind

Follow

We recently became aware of a potential data security issue. Protecting our members' info remains our top priority

[bit.ly/1PB3yox](http://bit.ly/1PB3yox)

8:53 PM - 22 May 2015

Retweets: 17 Stars: 4

**ASHLEY  
MADISON**<sup>®</sup>  
Life is short. Have an affair.<sup>®</sup>

Get started by telling us your relationship status:

Please Select

See Your Matches >

Over 37,665,000 anonymous members!



# Who's next to be hacked?



**OutFrontCNN** @OutFrontCNN · 7 Std.  
Researcher who hacked **Jeep Cherokee**: "We're the good guys"  
[cnnmon.ie/1Vx5BL4](http://cnnmon.ie/1Vx5BL4)



Researcher who hacked Jeep Cherokee: "We're the good guys"  
See more at [cnn.com](http://cnn.com)

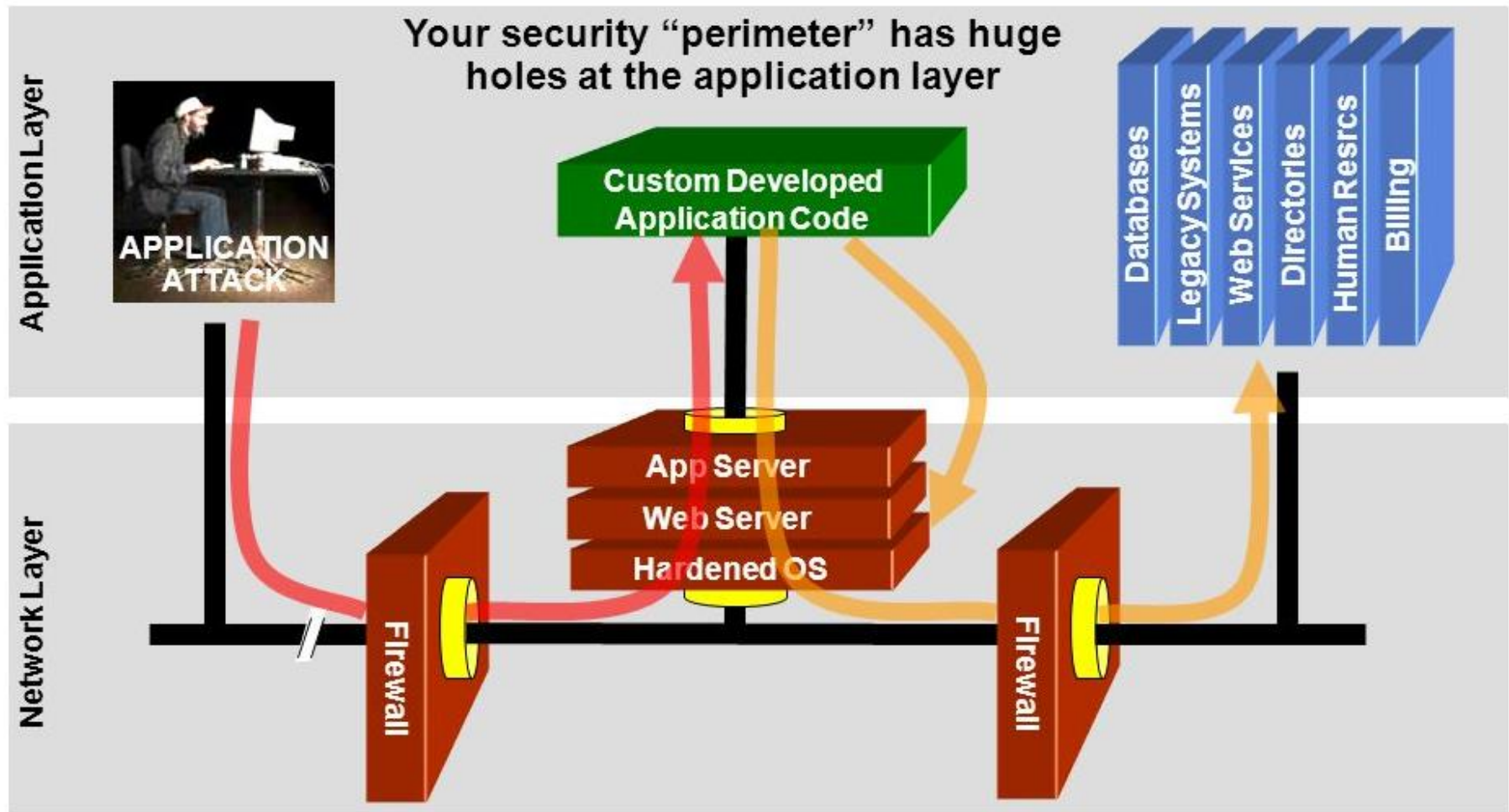
## HACKING AMERICA

### FBI: Computer expert briefly made plane fly sideways

Elizabeth Weise  
Sunday, 17 May 2015 | 11:45 AM ET



# Are firewalls really sufficient?



<http://owasp.org>



# Web Application Security: OWASP Top 10

**A1 – Injection**

**A2 – Broken Authentication and Session Management**

**A3 – Cross-Site Scripting (XSS)**

**A4 – Insecure Direct Object References**

**A5 – Security Misconfiguration**

**A6 – Sensitive Data Exposure**

**A7 – Missing Function Level Access Control**

**A8 – Cross-Site Request Forgery (CSRF)**

**A9 – Using Known Vulnerable Components**

**A10 – Unvalidated Redirects and Forwards**

**Merged with 2010-A7 into new 2013-A6**



**OWASP**

The Open Web Application Security Project

**OWASP Top 10 - 2013**

The Ten Most Critical Web Application Security Risks

release



Creative Commons (CC) Attribution Share-Alike  
Free version at <https://www.owasp.org>



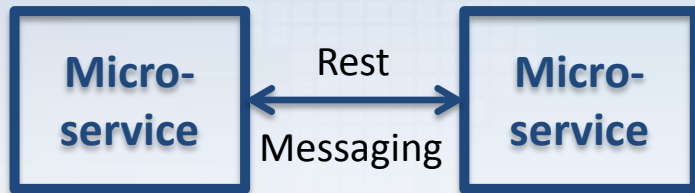
**OWASP**

Open Web Application  
Security Project

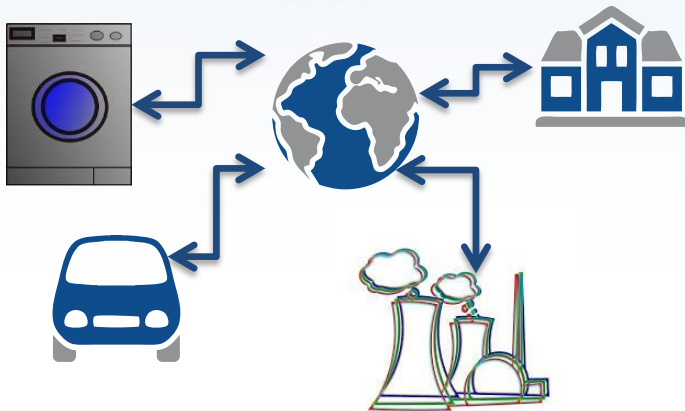
# New Security Challenges



Cloud Computing & Big Data



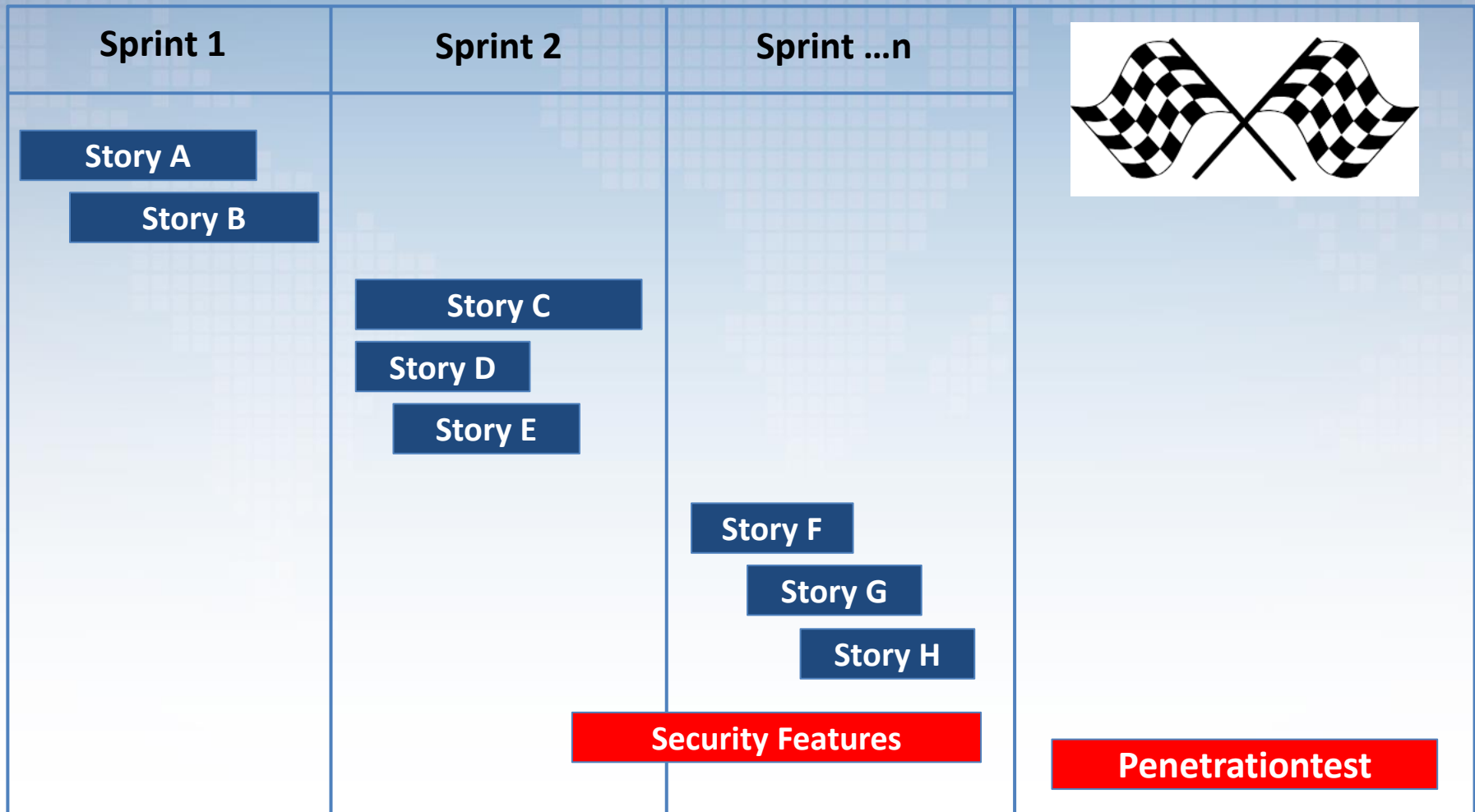
Microservices



Internet of Things (IoT)



# Security == Agile?



# Potentially Releasable?

## Scrum Guide:

“ The Development Team consists of professionals who do the work of delivering a potentially releasable increment of “Done” product at the end of each Sprint ”

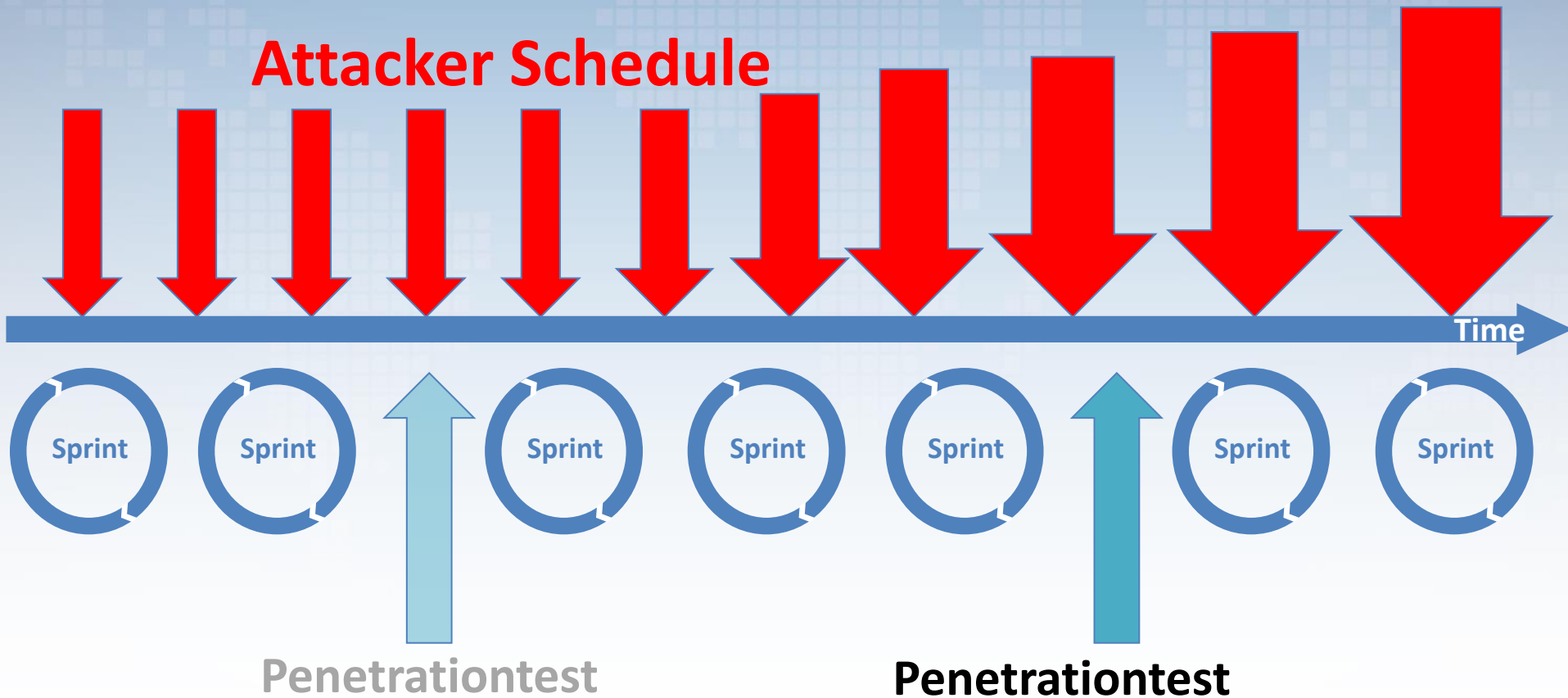
<http://www.scrumguides.org>



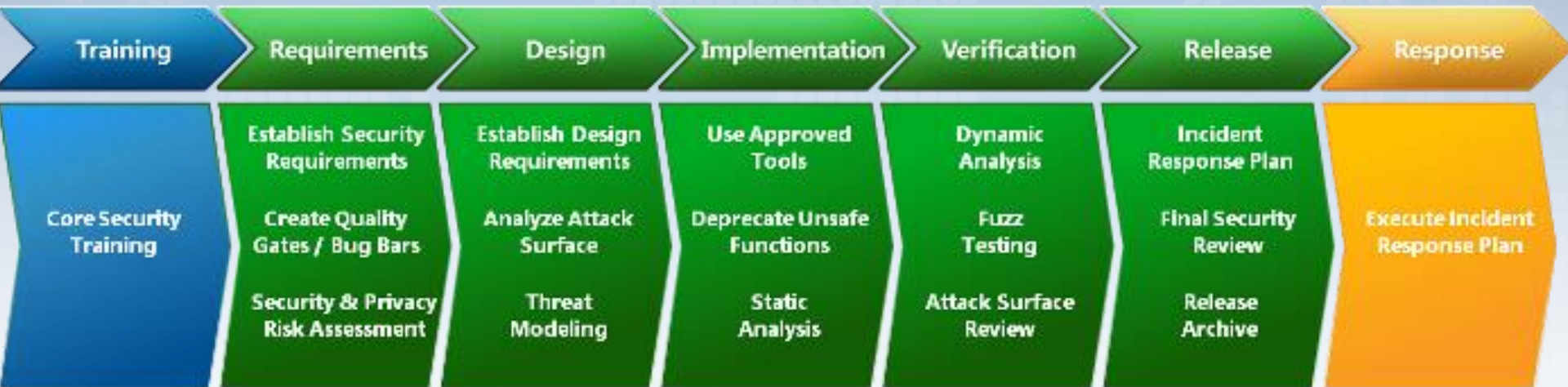
**Release potentially unsecure ?**

# Attacker Schedule: 24h x 7d

## Attacker Schedule

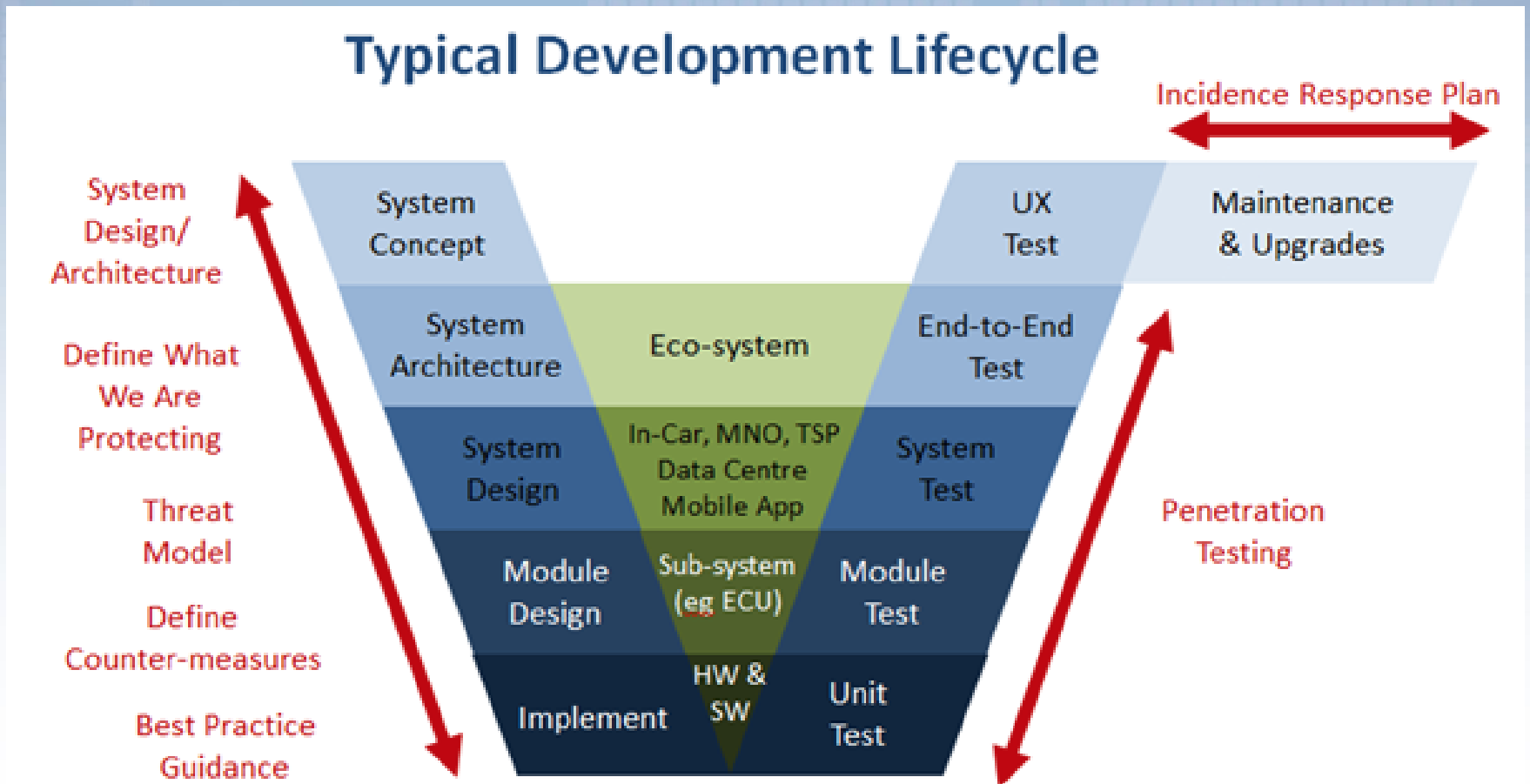


# Microsoft Security Development Lifecycle (SDL)



<https://www.microsoft.com/en-us/sdl/>

# Automotive SDL (V-Model)



# Next Stop: **Secure** Agile Development



# Manifesto for Agile Software Development

**Individuals and Interactions** over processes and tools

**Working software** over comprehensive documentation

**Customer collaboration** over contract negotiation

**Responding to change** over following a plan

# Manifesto for **Secure** Agile Software Development

**Individuals and Interactions** over ~~processes and tools~~ **complex security policies**

**Secure working software**  
~~Working software~~ over comprehensive documentation

**Customer collaboration** over ~~contract negotiation~~ **vague security requirements**

**Responding to change** over following a ~~plan~~ **static threat model**



# Agile Development



# The Rugged Manifesto

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.

I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic, and national security.

I am rugged because I refuse to be a source of vulnerability or weakness.

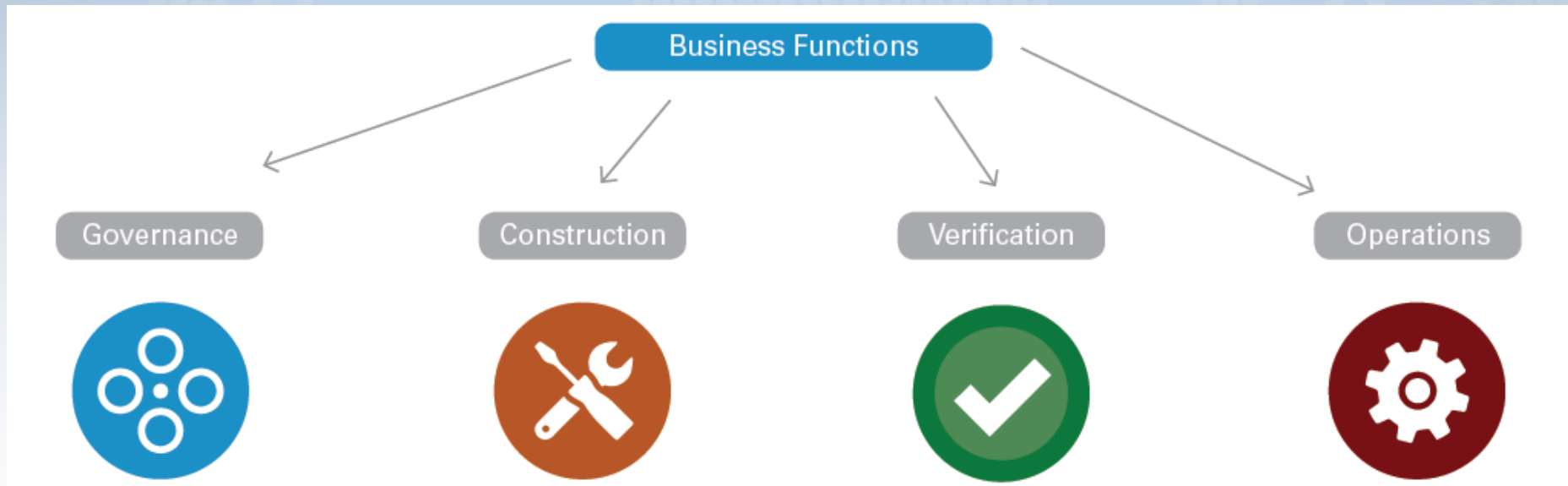
I am rugged, not because it is easy, but because it is necessary... and I am up for the challenge.

<https://www.ruggedsoftware.org>

# Microsoft SDL for Agile Development

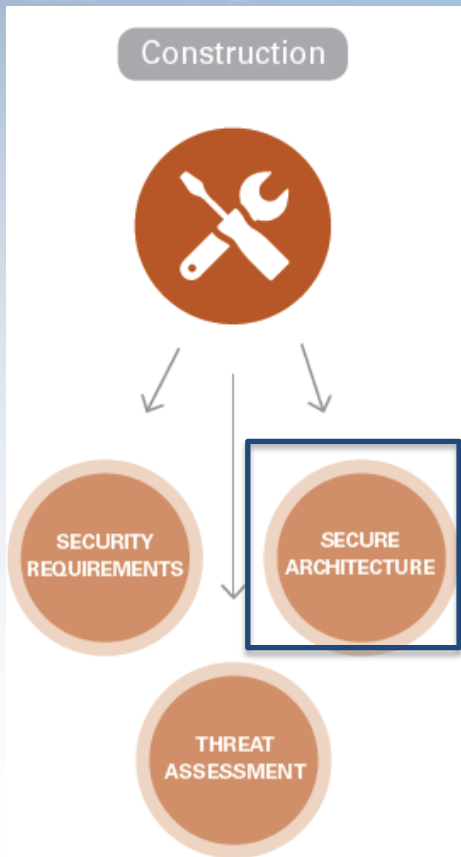
Agile Process	SDLC Tasks
<b>One Time</b>	<ul style="list-style-type: none"><li>• Baseline threat model</li><li>• Establish security response plan</li><li>• ...</li></ul>
<b>Regular Basis</b>	<ul style="list-style-type: none"><li>• Privacy review</li><li>• Manual &amp; automatic security code review</li><li>• ...</li></ul>
<b>Every Sprint</b>	<ul style="list-style-type: none"><li>• Security training</li><li>• Threat modeling</li><li>• Secure coding</li><li>• Code reviews</li><li>• ...</li></ul>

# OWASP Open Software Assurance Maturity Model (OpenSAMM)



<http://www.opensamm.org/>

# OpenSAMMM - Construction



- Are project teams provided with a list of recommended third-party components?
- Are most project teams aware of secure design principles and applying them?

SA

1

- Do you advertise shared security services with guidance for project teams?
- Are project teams provided with prescriptive design patterns based on their application architecture?

SA

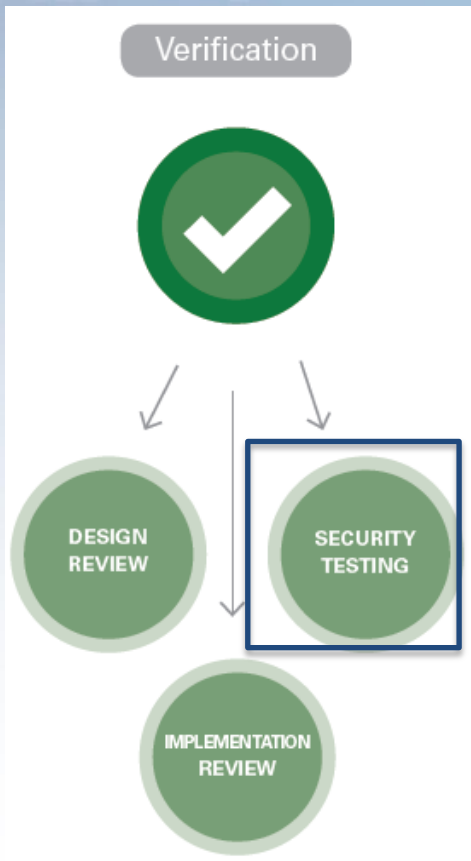
2

- Are project teams building software from centrally controlled platforms and frameworks?
- Are project teams being audited for usage of secure architecture components?

SA

3

# OpenSAMM - Verification



- Are projects specifying some security tests based on requirements?
- Do most projects perform penetration tests prior to release?
- Are most stakeholders aware of the security test status prior to release?

ST

1

- Are projects using automation to evaluate security test cases?
- Do most projects follow a consistent process to evaluate and report on security tests to stakeholders?

ST

2

- Are security test cases comprehensively generated for application-specific logic?
- Do routine project audits demand minimum standard results from security testing?

ST

3

# Agile Development with Scrum

Developer Team



Product Owner



Scrum Master

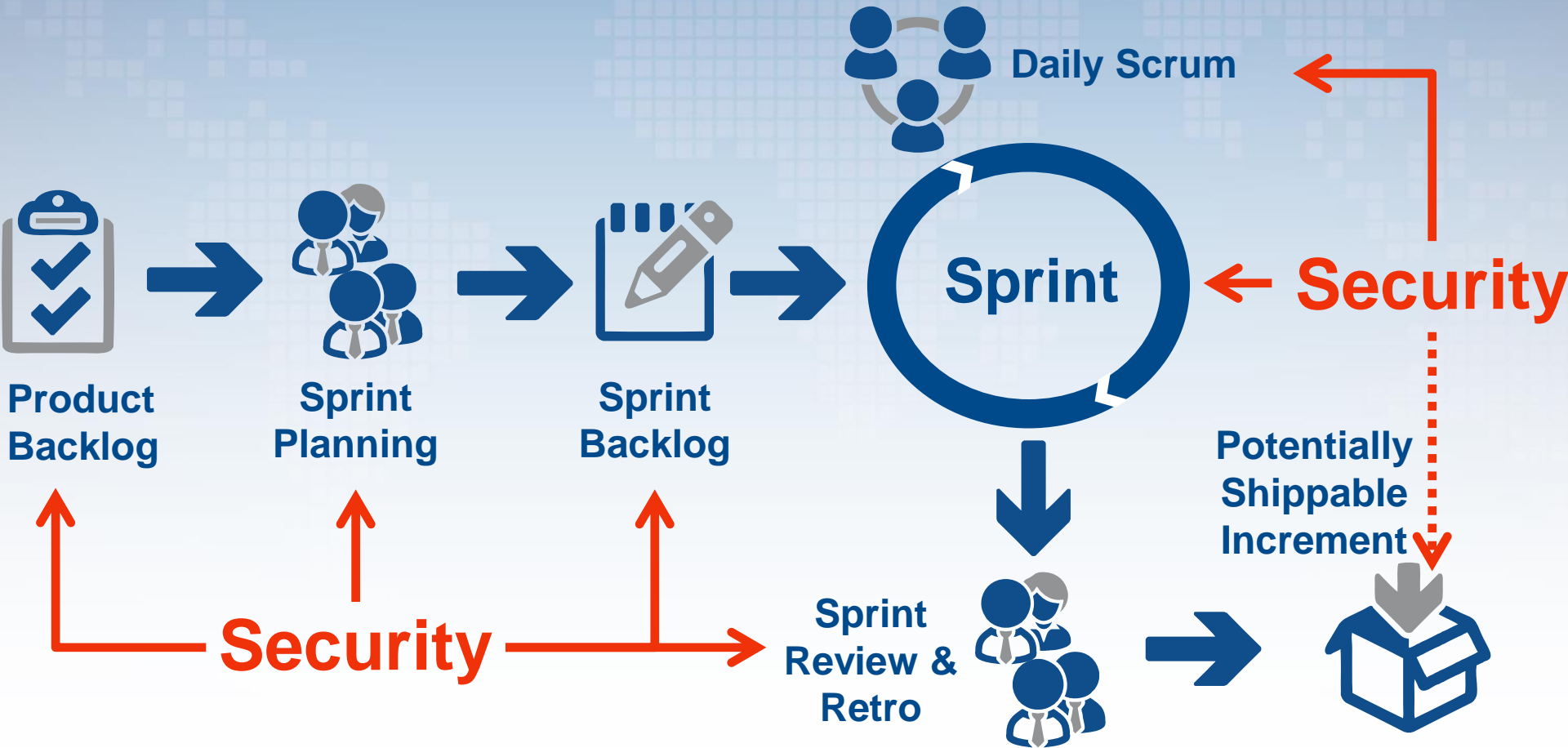


Developer



Test & QA

# Secure Agile Development with Scrum



✓ **Security-Trainings**



# Security Trainings

Product Owner



- Security Risk Identification & Management
- Data Privacy Requirements
- Threat Modeling
- Security Features
- Security User Stories („Abuse Stories“)

Development  
Team



- Threat Modeling
- Secure Coding
- Security Code Reviews
- Security Testing (Manual & Automatic)

# Threat Modeling is „Agile“...

## Create Production Code

Make Tests Pass

## Test Driven Development (TDD)

Write Security Tests First

## Create Security Testcases and Abuse User Stories



## Define Software-Architecture

User Stories,  
UML Diagrams

## Adapt Threat Model

Discussion Basis

## Identify and Mitigate Threats

„Elevation of privilege“ game

# Playing games...



**Scrum Planning Poker**



**Threat Modeling Game**

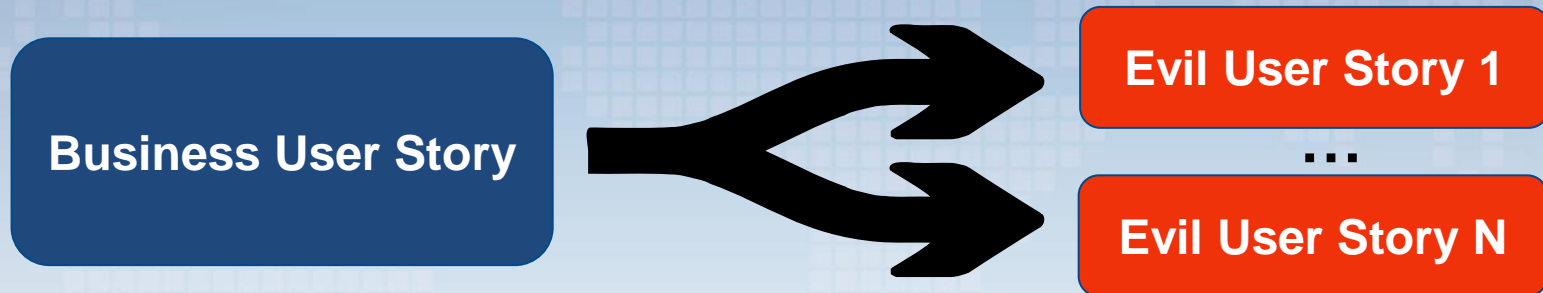
# Secure Agile Development with Scrum



**Product Backlog**

- Update threat model (on-going)
- Define abuse user stories
- Plan security features early
- Security acceptance criteria
- Extend „Definition of Ready“ with security

# Abuse (Evil) User Stories



As a customer I want to select products and add them to my shopping cart in order to buy these.

As an evil user I want to manipulate requests to change prices when adding products to my shopping cart.

# Secure Agile Development with Scrum



## Sprint Planning

- Update threat model (on-going)
- Plan abuse user story tasks
- Plan security features tasks
- Refine security acceptance criteria
- Define and plan security test strategies

# Secure Agile Development with Scrum



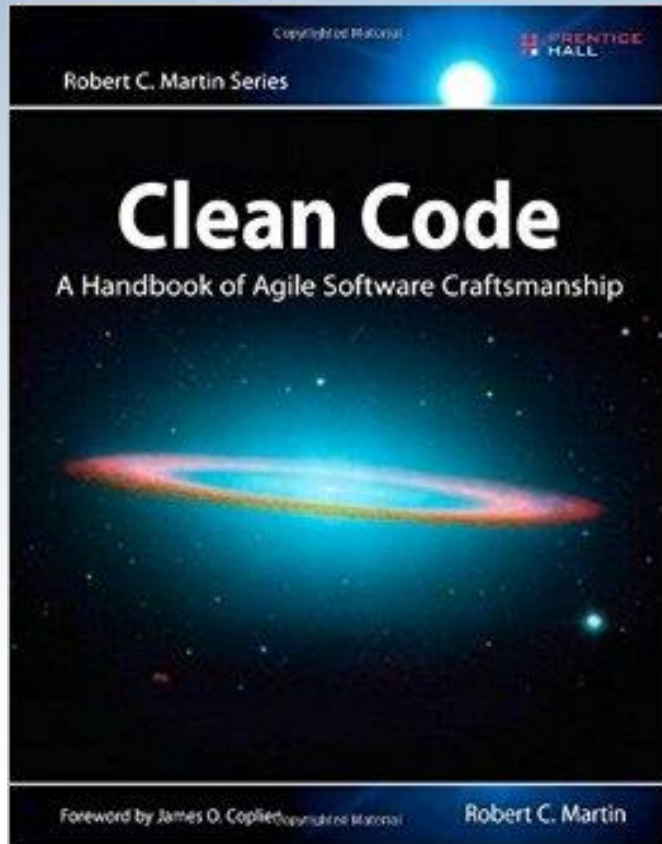
Daily Scrum

- Discuss security risks
- (Re-)plan security tasks

- Secure coding
- Pair programming/test with security expert
- „Security Officer“ developer
- „Security-Aware“ Definition of Done
- Security regression testing (CI)
- Security code reviews



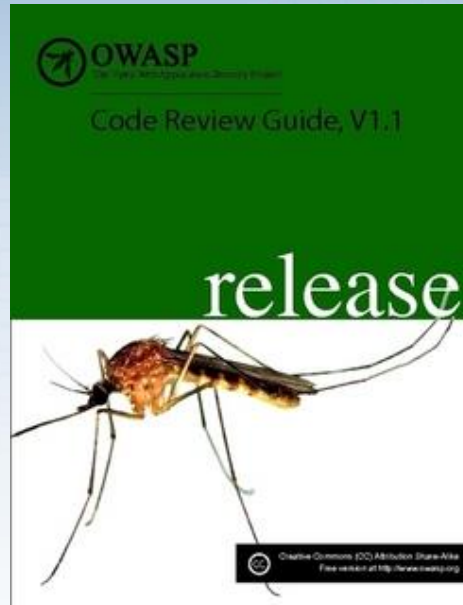
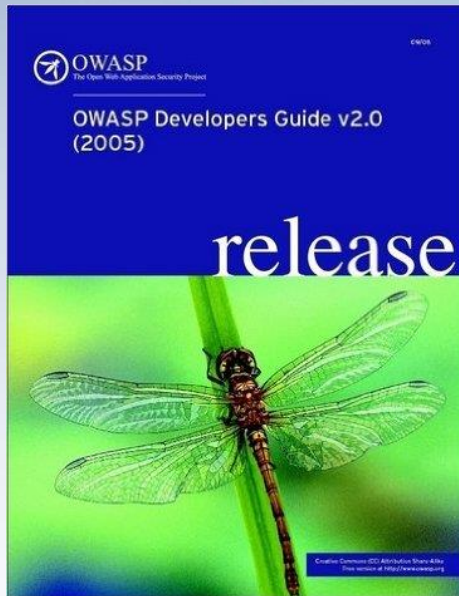
# Secure Coding – „Clean Code“



- Boundaries (3rd party code)
- Error Handling
- Good Comments
- Unit Tests
- Constant Refactoring
- Keep it simple & DRY
- ...



# Secure Coding – Security Patterns



- Input validation
- Output escaping
- Use prepared SQL statements
- No sensible data in logs
- No stack traces on the UI
- Session management
- Access Controls
- ...



# Secure Coding – Standard Frameworks



Custom Enterprise Web Application

Enterprise Security API

Authenticator

User

AccessController

AccessReferenceMap

Validator

Encoder

HTTPUtilities

Encryptor

EncryptedProperties

Randomizer

Exception Handling

Logger

IntrusionDetector

SecurityConfiguration

- Standard cryptography
- Validating/escaping UI frameworks e.g.
  - JavaServer Faces
  - Vaadin (GWT)
- Proven security frameworks
  - Spring Security
  - Apache Shiro
  - OWASP ESAPI

# Secure Coding – Secure app in 5 minutes

New Project

Spring Boot Version: 1.2.5

Dependencies

Support for spring-security

- Security
- Cache
- AOP
- DevTools
- Atomikos (JTA)
- Bitronix (JTA)

Web

- Web
- Vaadin
- Websocket
- Rest Repositories
- WS
- HATEOAS
- Jersey (JAX-RS)
- Mobile

Template Engines

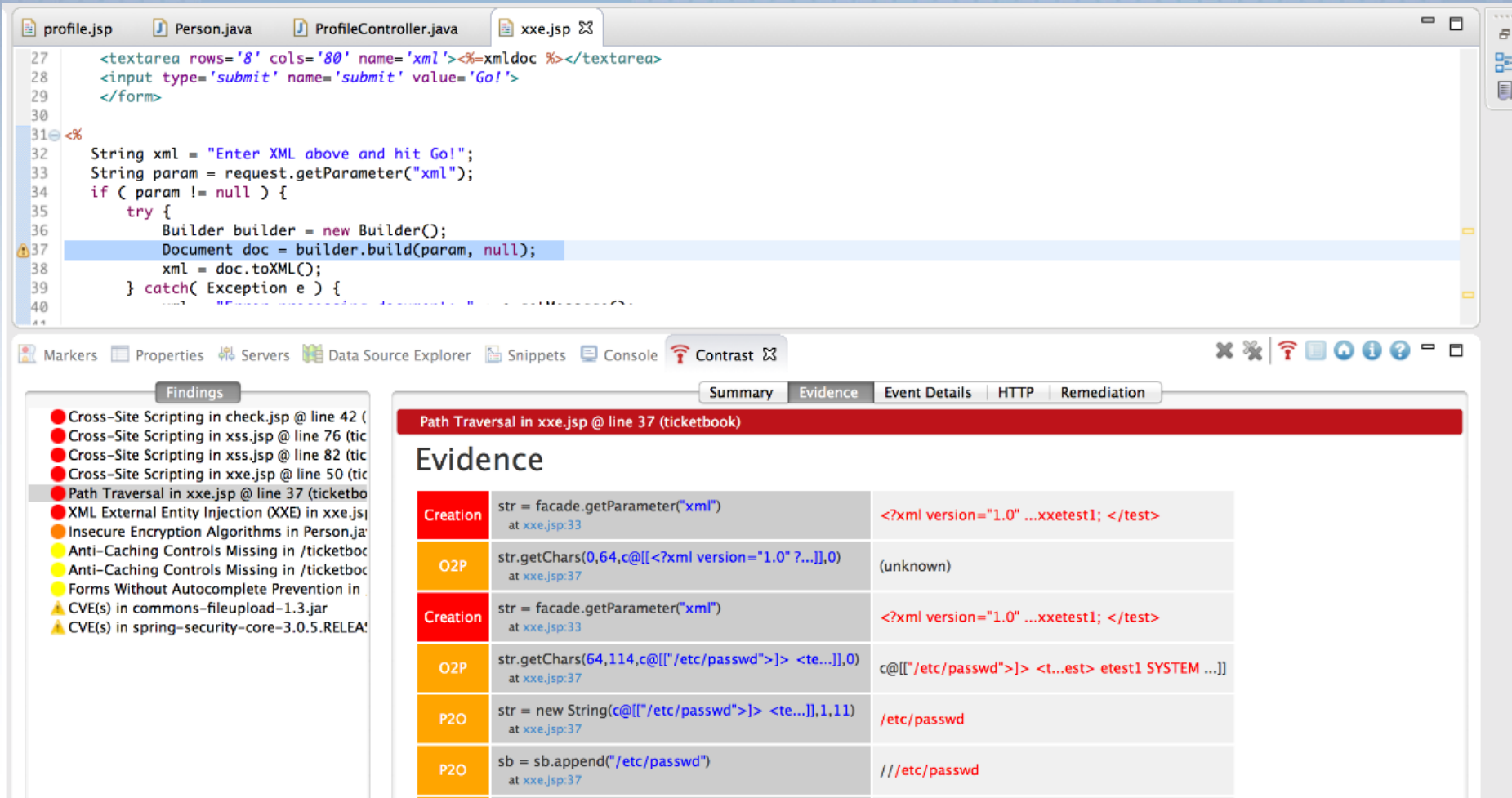
- Freemarker
- Mustache
- Velocity
- Groovy Templates
- Thymeleaf

Data

- JDBC
- Gemfire
- JPA
- Solr
- MongoDB
- Elasticsearch
- Redis

Previous Next Cancel Help

# Secure Coding – Security issues in the IDE



```
27 <textarea rows='8' cols='80' name='xml'><%=xml doc %></textarea>
28 <input type='submit' name='submit' value='Go!'>
29 </form>
30
31 <%=
32 String xml = "Enter XML above and hit Go!";
33 String param = request.getParameter("xml");
34 if ( param != null ) {
35     try {
36         Builder builder = new Builder();
37         Document doc = builder.build(param, null);
38         xml = doc.toXML();
39     } catch( Exception e ) {
40         ...
41     }
42 }
```

**Findings**

- Cross-Site Scripting in check.jsp @ line 42 (
- Cross-Site Scripting in xss.jsp @ line 76 (tic
- Cross-Site Scripting in xss.jsp @ line 82 (tic
- Cross-Site Scripting in xxe.jsp @ line 50 (tic
- Path Traversal in xxe.jsp @ line 37 (ticketbo
- XML External Entity Injection (XXE) in xxe.js
- Insecure Encryption Algorithms in Person.ja
- Anti-Caching Controls Missing in /ticketboc
- Anti-Caching Controls Missing in /ticketboc
- Forms Without Autocomplete Prevention in
- ▲ CVE(s) in commons-fileupload-1.3.jar
- ▲ CVE(s) in spring-security-core-3.0.5.RELEA

**Path Traversal in xxe.jsp @ line 37 (ticketbook)**

**Evidence**

Severity	Code Snippet	Output
Creation	str = facade.getParameter("xml") at xxe.jsp:33	<?xml version="1.0" ...xxetest1; </test>
O2P	str.getChars(0,64,c@[["<?xml version="1.0" ?...]],0) at xxe.jsp:37	(unknown)
Creation	str = facade.getParameter("xml") at xxe.jsp:33	<?xml version="1.0" ...xxetest1; </test>
O2P	str.getChars(64,114,c@[["/etc/passwd"]> <te...]],0) at xxe.jsp:37	c@[["/etc/passwd"]> <t...est> etest1 SYSTEM ...]
P2O	str = new String(c@[["/etc/passwd"]> <te...]],1,11) at xxe.jsp:37	/etc/passwd
P2O	sb = sb.append("/etc/passwd") at xxe.jsp:37	///etc/passwd

<http://www.contrastsecurity.com/eclipse>



8/3/2015

Chapter Meeting Stuttgart

# Secure Agile Testing

Business Facing

Automated & Manual

Manual

Supporting the Team

Functional Tests  
Examples  
Story Tests  
Prototypes  
Simulations

Q2

Q3

Exploratory Testing  
Scenarios  
Usability Testing  
UAT (User Acpt. Testing)  
Alpha / Beta

Critique Product

Q1

Q4

Unit Tests  
Component Tests

Performance &  
Load Testing  
Security Testing  
„ility“ Testing

Automated

Technology Facing

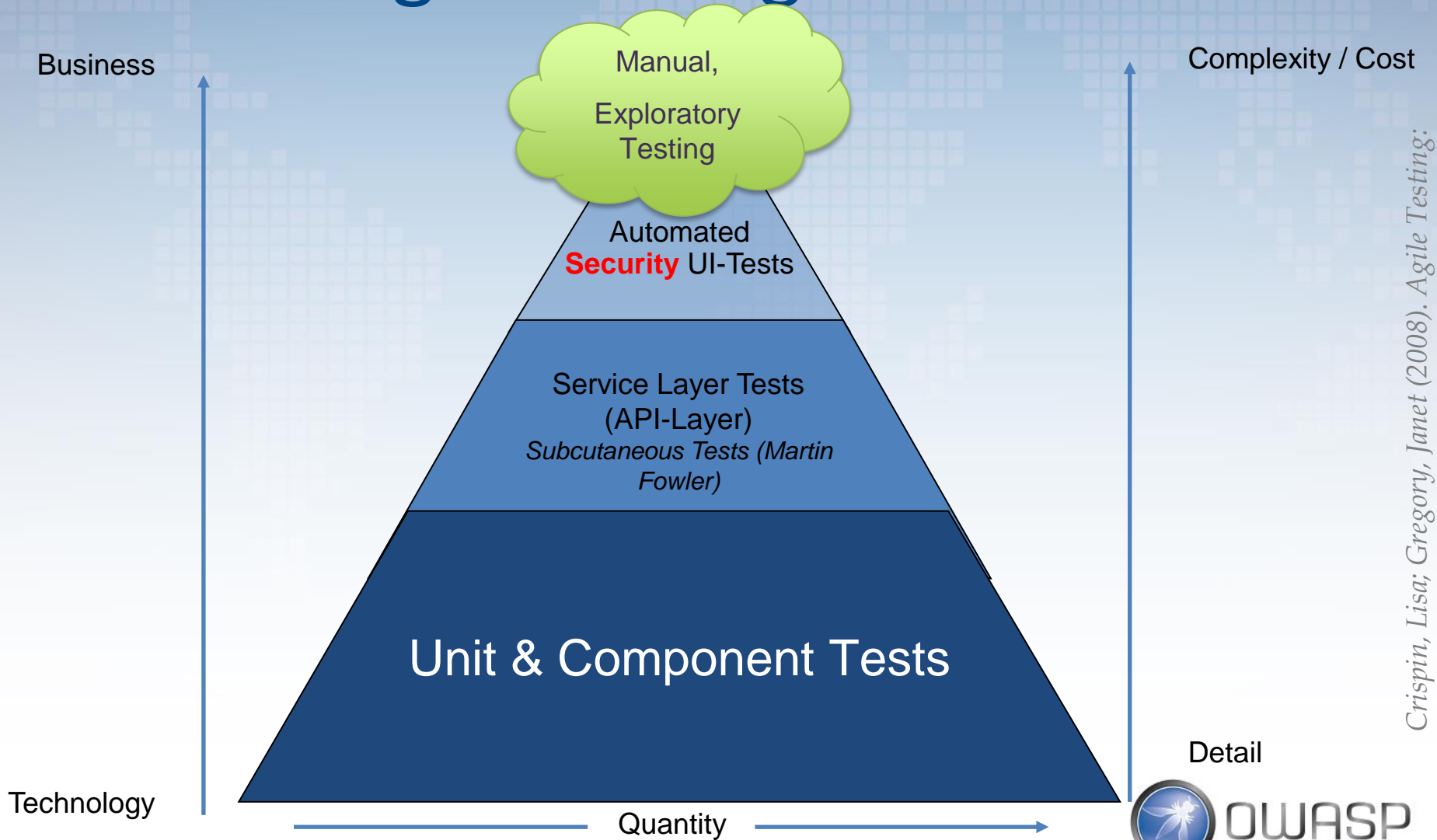
Tools

*Crispin, Lisa; Gregory, Janet (2008). Agile Testing:*



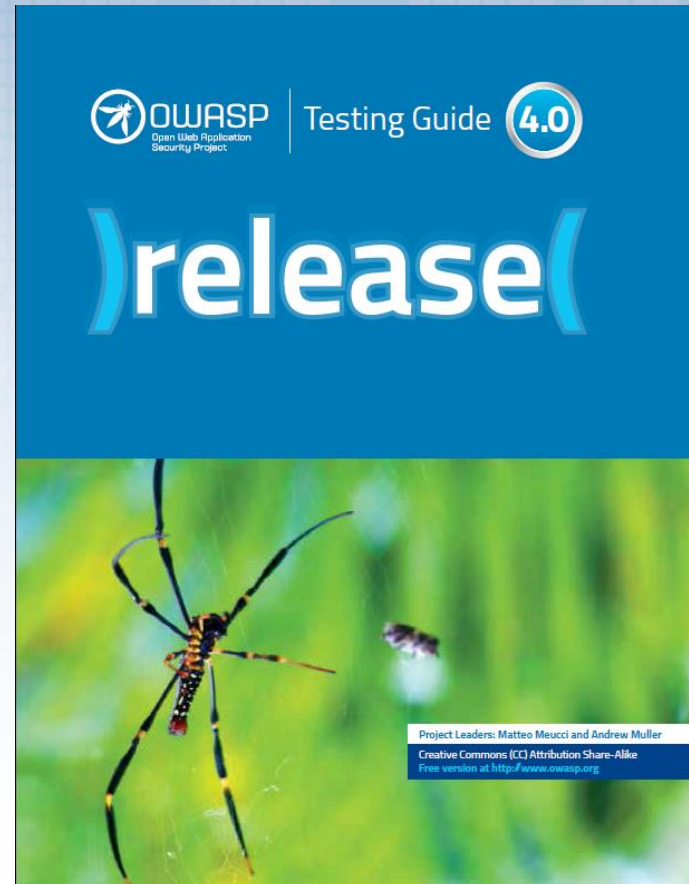
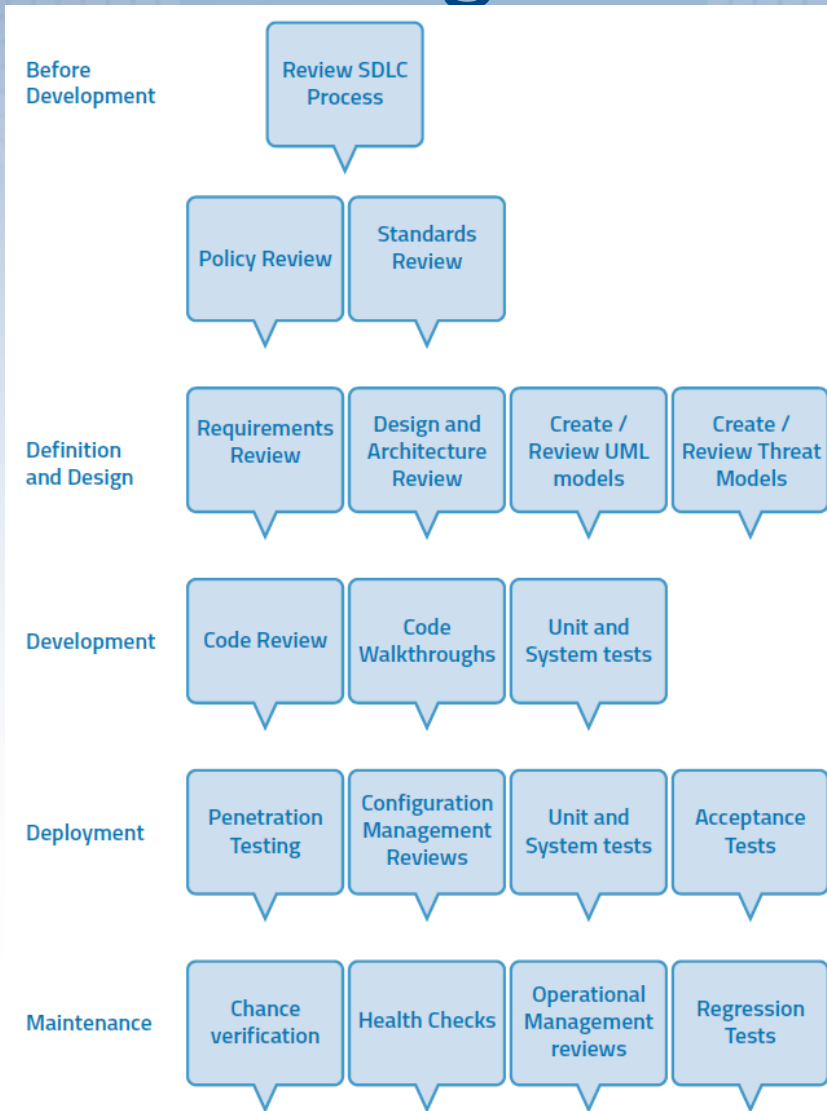
**OWASP**  
Open Web Application  
Security Project

# Secure Agile Testing



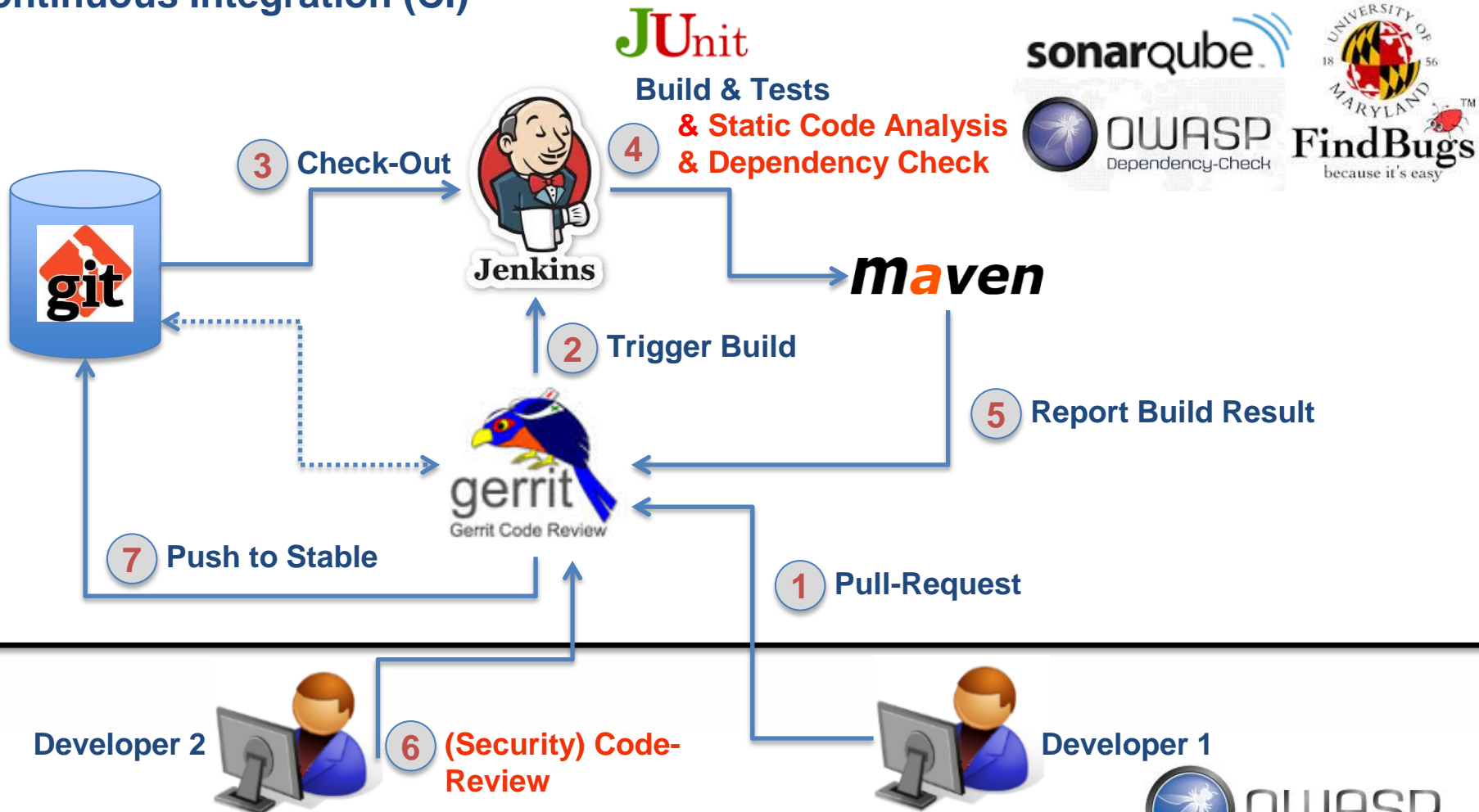
*Crispin, Lisa; Gregory, Janet (2008). Agile Testing:*

# Secure Agile Testing



# Stage 1: Static Security Testing

## Continuous Integration (CI)



Developer 2



6 (Security) Code-Review



Developer 1





# Static Code Analysis: FindBugs



+



## Find Security Bugs

The FindBugs plugin for security audits of Java web applications.

- Detects 63 different vulnerability patterns
- OWASP TOP 10 and CWE coverage
- IDE- and CI-integrations

<http://findbugs.sourceforge.net>

<http://h3xstream.github.io/find-sec-bugs>

# Static Code Analysis: SonarQube

## + Java Plugin

- Detects lots of security problems
  - SANS TOP 25
  - OWASP Top 10
  - CWE references (<http://cwe.mitre.org>)

<http://www.sonarqube.org>

<http://docs.sonarqube.org/display/PLUG/Java+Plugin>

# Dependency Check

- Detects vulnerabilities in 3rd party libraries
  - Java, .NET, Python applications
  - Checks library information against imported CVE data
  - Sometimes needs suppressing false positives
  - Supports Command line, Ant, Maven, Gradle, Jenkins

## Project: Bodgeit Store Vulnerable Application

Scan Information ([show all](#)):

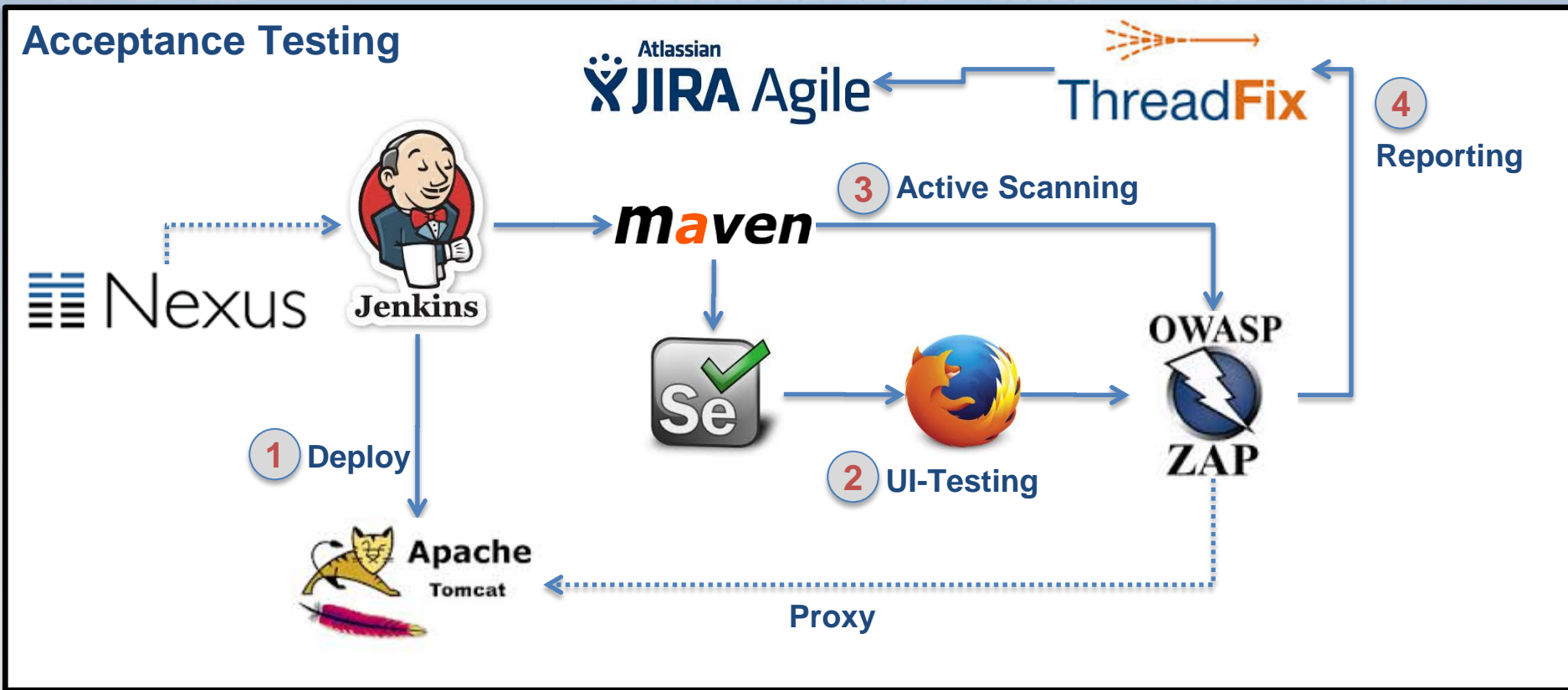
- *dependency-check* version: 1.2.11
- *Report Generated On*: Aug 2, 2015 at 23:32:36 MESZ
- *Dependencies Scanned*: 3
- *Vulnerable Dependencies*: 0
- *Vulnerabilities Found*: 0
- *Vulnerabilities Suppressed*: 0
- ...

Display: [Showing All Dependencies \(click to show less\)](#)

Dependency	CPE	GAV	Highest Severity	CVE Count	CPE Confidence	Evidence Count
<a href="#">javax.servlet-api-3.1.0.jar</a>		<a href="#">javax.servlet:javax.servlet-api:3.1.0</a>		0		21
<a href="#">commons-lang3-3.3.2.jar</a>		<a href="#">org.apache.commons:commons-lang3:3.3.2</a>		0		24
<a href="#">hsqldb-2.3.2.jar</a>		<a href="#">org.hsqldb:hsqldb:2.3.2</a>		0		21

<http://jeremylong.github.io/DependencyCheck>

# Stage 2: Dynamic Security-Testing



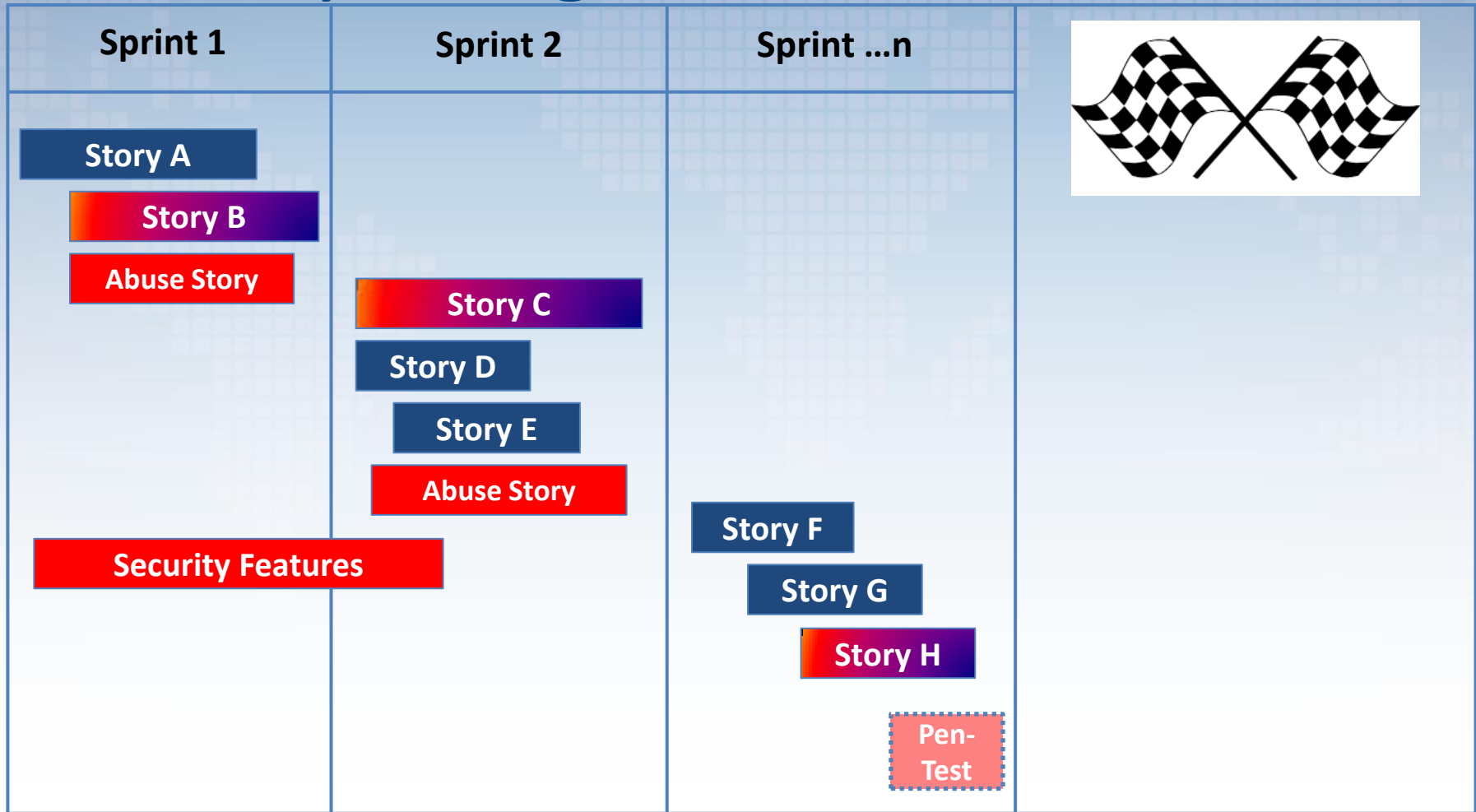
# Secure Agile Development with Scrum



## Sprint Review & Retro

- Verify threat model coverage
- Review abuse user stories and security acceptance criteria
- Make security topics visible for customer
- Inspect and adapt security activities

# Security == Agile!



# Don't make it that EASY to break software!

