PCI-DSS v1.2 and OWASP

Matteo Meucci

OWASP-Italy Chair
OWASP Testing Guide Lead
CISA, CISSP

PCI Milan 09
March 31st, 2009 - Italy

Agenda:

- Introduction to OWASP
- PCI-DSS Standard & OWASP
- The OWASP Testing Guide
- Q&A

Who am I?

- Research
  - OWASP-Italy Chair
  - OWASP Testing Guide Lead

- Work
  - CEO @ Minded Security

Application Security Consulting

  - 8+ years on Information Security

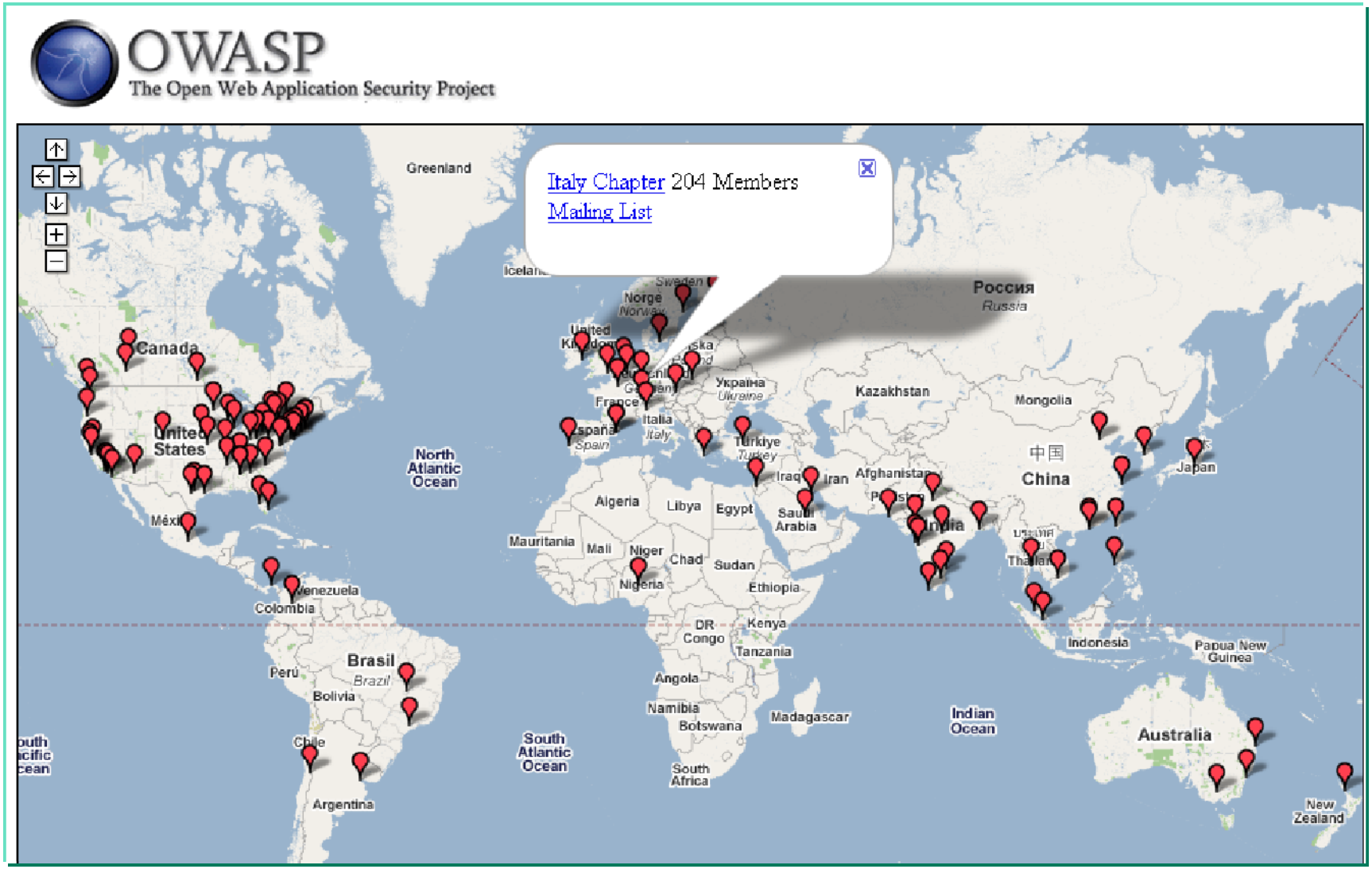
focusing on Application Security

  - www.mindedsecurity.com

The Open Web Application Security Project

- The Open Web Application Security Project (OWASP) is dedicated to finding and fighting the causes of insecure software. The OWASP Foundation is a 501c3 not-for-profit charitable organization that ensures the ongoing availability and support for our work.
- Participation in OWASP is free and open to all.
- Everything here is free and open source.
- Main objectives: producing tools, standards and documentations related to Web Application Security.
- Thousands active members, 130+ local chapters in the world
- Millions of hits on www.owasp.org

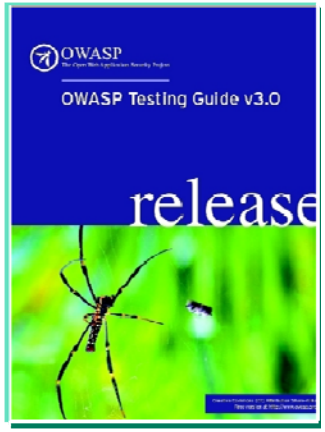
The OWASP Community



OWASP v3: Improve Quality and Support

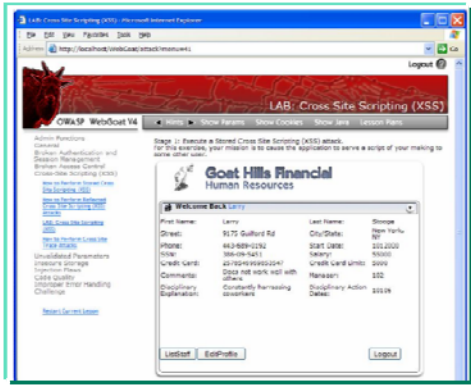
- Define Criteria for Quality Levels
  - Alpha, Beta, Release
- Encourage Increased Quality
  - Through Season of Code Funding and Support
  - Produce Professional OWASP books
- Provide Support
  - Full time executive director (Kate Hartmann)
  - Full time project manager (Paulo Coimbra)
  - Half time technical editor (Kirsten Sitnick)
  - Half time financial support (Alison Shrader)
  - Looking to add programmers (Interns and professionals)

OWASP Release projects



BOOKS

- Owasp top10
- Building guide
- Code review guide
- Testing guide
- Back end security

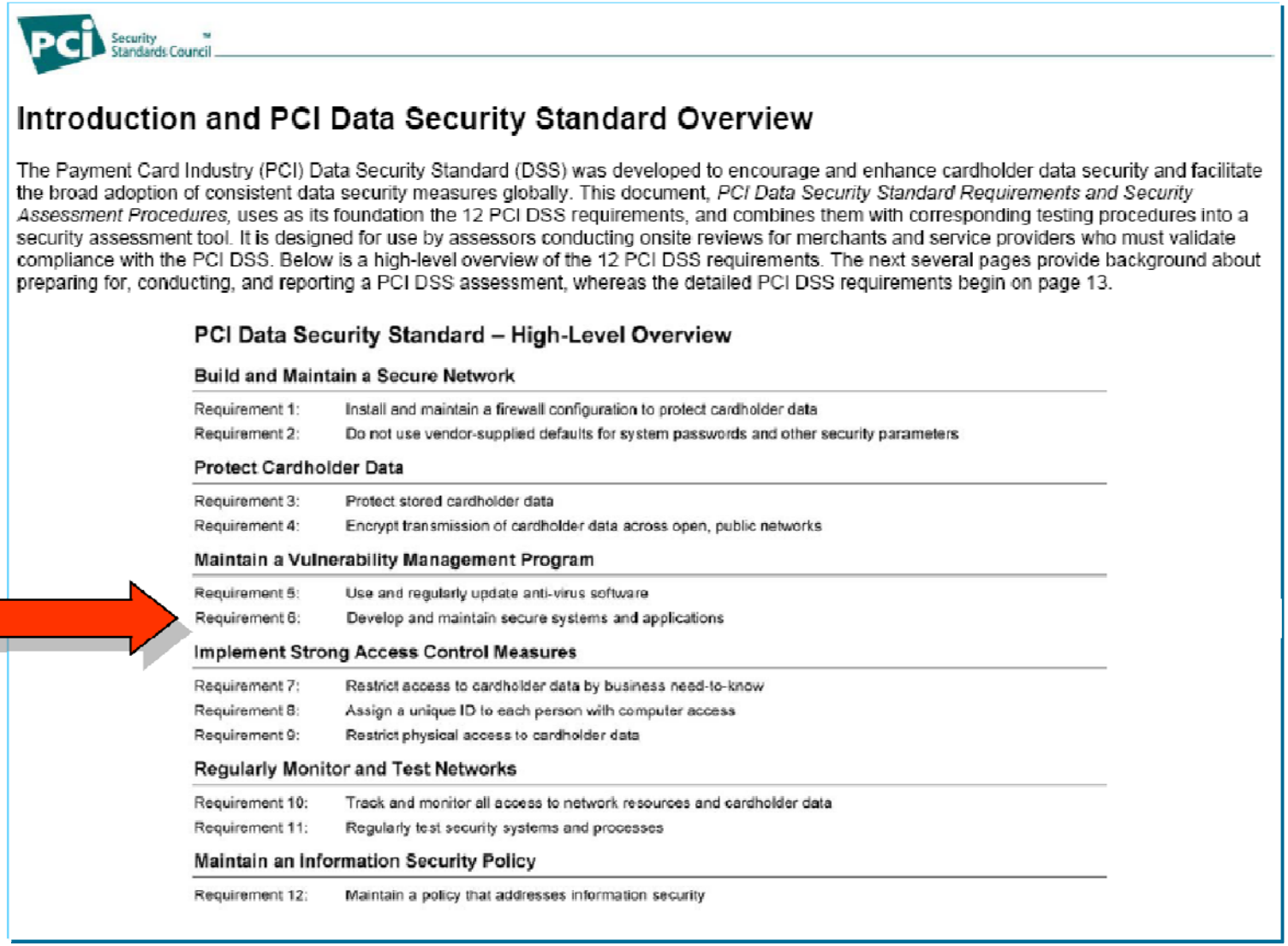


TOOLS

- WebGoat
- WebScarab
- SQLMap – SQL Ninja
- SWF Intruder
- Orizon
- Code Crawler

PCI-DSS v1.2 and OWASP

PCI-DSS: focus on Application Security



PCI Security Standards Council

Introduction and PCI Data Security Standard Overview

The Payment Card Industry (PCI) Data Security Standard (DSS) was developed to encourage and enhance cardholder data security and facilitate the broad adoption of consistent data security measures globally. This document, *PCI Data Security Standard Requirements and Security Assessment Procedures*, uses as its foundation the 12 PCI DSS requirements, and combines them with corresponding testing procedures into a security assessment tool. It is designed for use by assessors conducting onsite reviews for merchants and service providers who must validate compliance with the PCI DSS. Below is a high-level overview of the 12 PCI DSS requirements. The next several pages provide background about preparing for, conducting, and reporting a PCI DSS assessment, whereas the detailed PCI DSS requirements begin on page 13.

PCI Data Security Standard – High-Level Overview

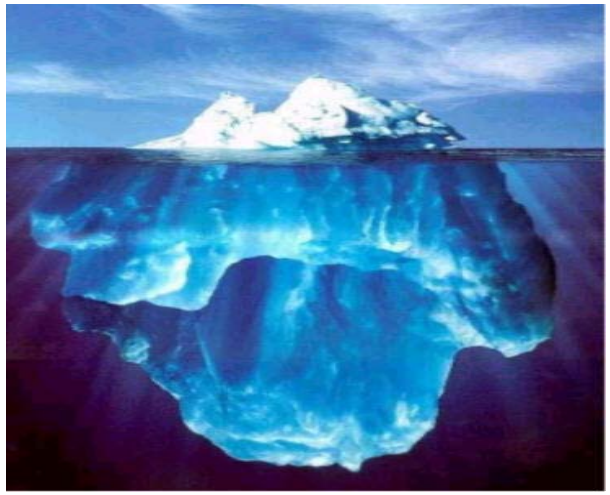
Build and Maintain a Secure Network	
Requirement 1:	Install and maintain a firewall configuration to protect cardholder data
Requirement 2:	Do not use vendor-supplied defaults for system passwords and other security parameters
Protect Cardholder Data	
Requirement 3:	Protect stored cardholder data
Requirement 4:	Encrypt transmission of cardholder data across open, public networks
Maintain a Vulnerability Management Program	
Requirement 5:	Use and regularly update anti-virus software
Requirement 6:	Develop and maintain secure systems and applications
Implement Strong Access Control Measures	
Requirement 7:	Restrict access to cardholder data by business need-to-know
Requirement 8:	Assign a unique ID to each person with computer access
Requirement 9:	Restrict physical access to cardholder data
Regularly Monitor and Test Networks	
Requirement 10:	Track and monitor all access to network resources and cardholder data
Requirement 11:	Regularly test security systems and processes
Maintain an Information Security Policy	
Requirement 12:	Maintain a policy that addresses information security

PCI v1.2 & OWASP

6.5 Develop all web applications (internal and external, and including web administrative access to application) based on secure coding guidelines such as the *Open Web Application Security Project Guide*. Cover prevention of common coding vulnerabilities in software development processes, to include the following: *Note: The vulnerabilities listed at 6.5.1 through 6.5.10 were current in the OWASP guide when PCI DSS v1.2 was published. However, if and when the OWASP guide is updated, the current version must be used for these requirements.*	**6.5.a** Obtain and review software development processes for any web-based applications. Verify that processes require training in secure coding techniques for developers, and are based on guidance such as the OWASP guide (http://www.owasp.org).
	6.5.b Interview a sample of developers and obtain evidence that they are knowledgeable in secure coding techniques.
	6.5.c Verify that processes are in place to ensure that web applications are not vulnerable to the following:

PCI DSS Requirements and Security Assessment Procedures, v1.2
Copyright 2008 PCI Security Standards Council LLC

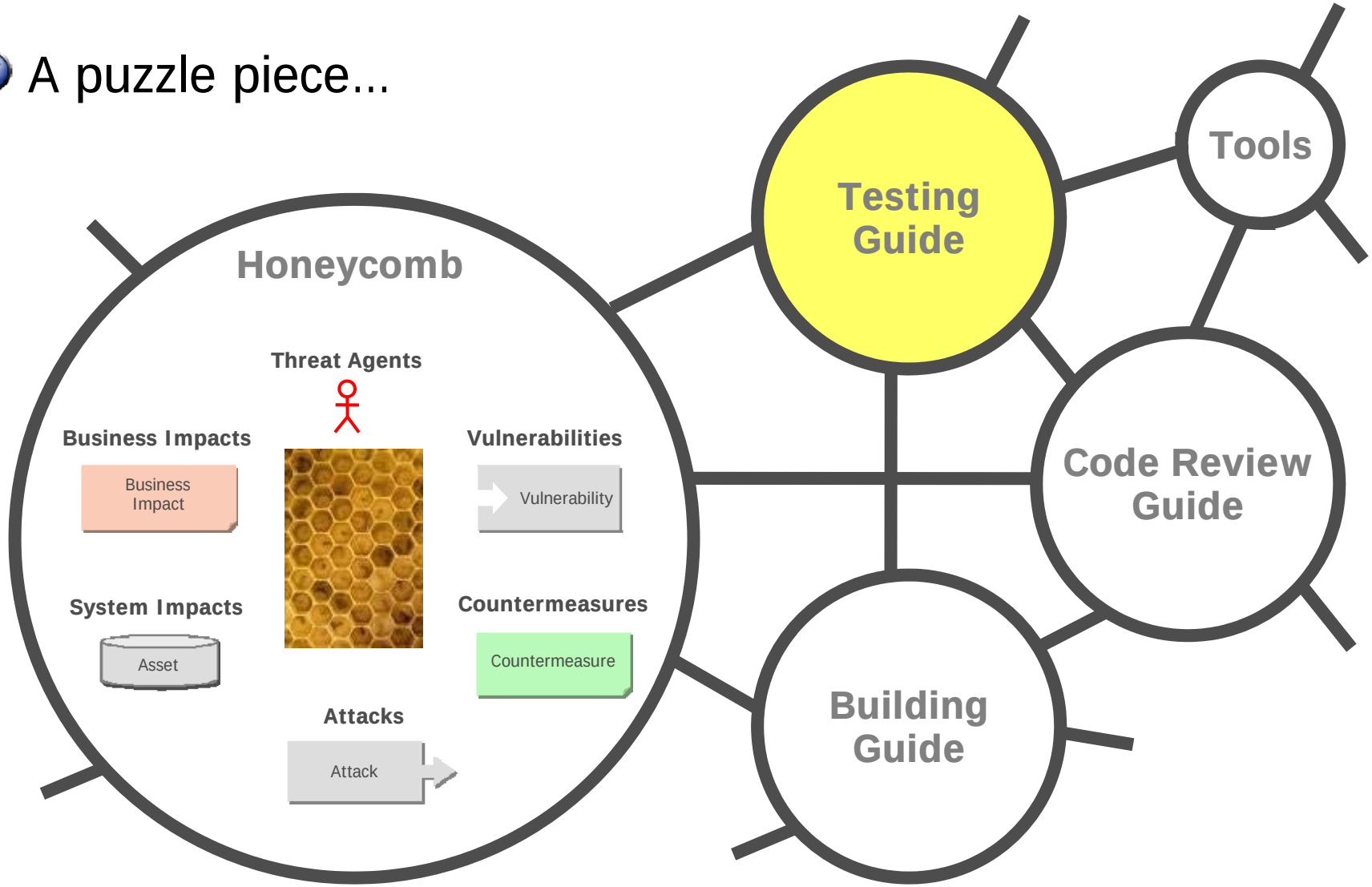
OWASP Top 10



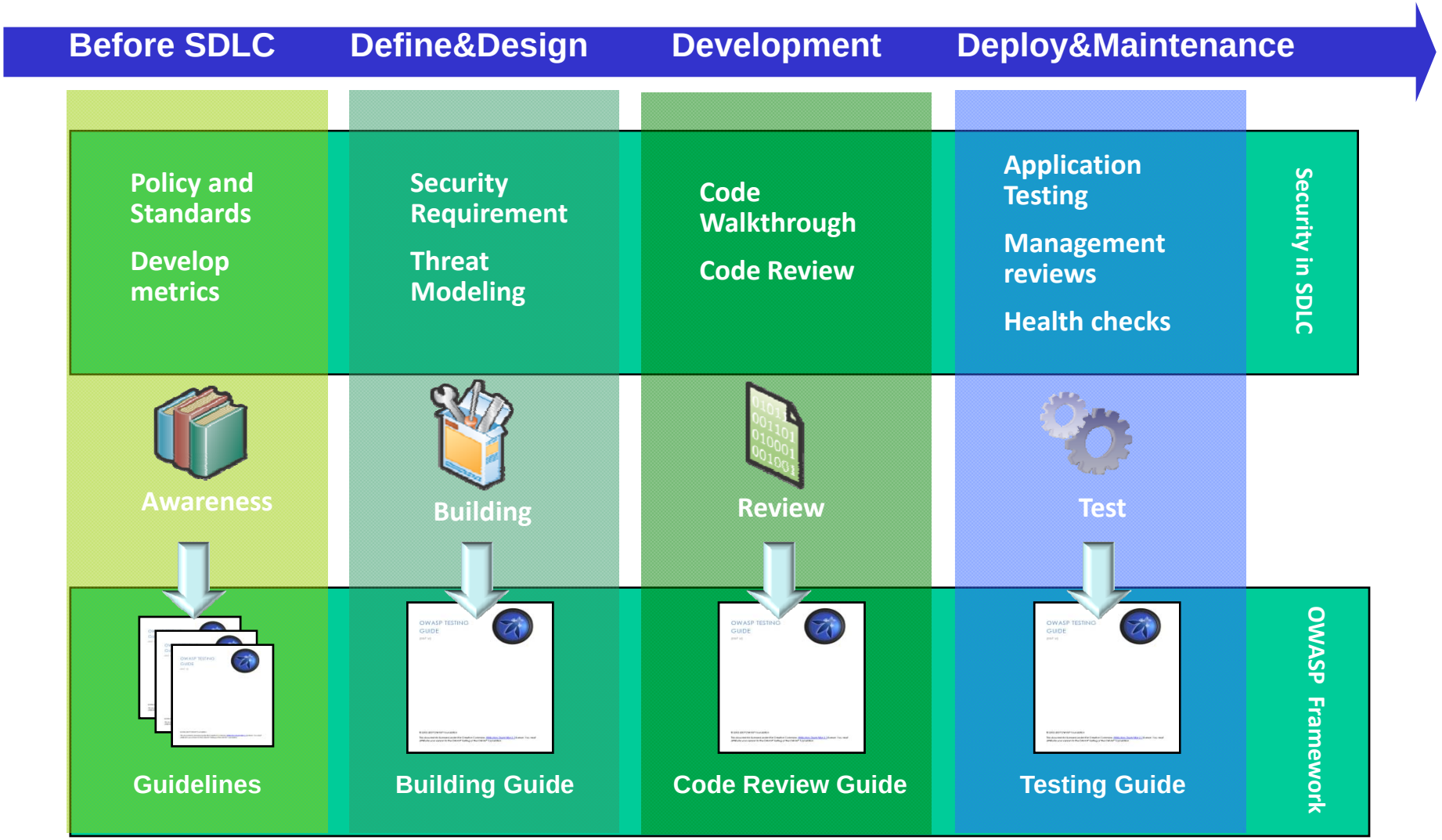
- The Ten Most Critical Web Application Security Vulnerabilities
- A great start, but not a standard

OWASP Guidelines

- A puzzle piece...

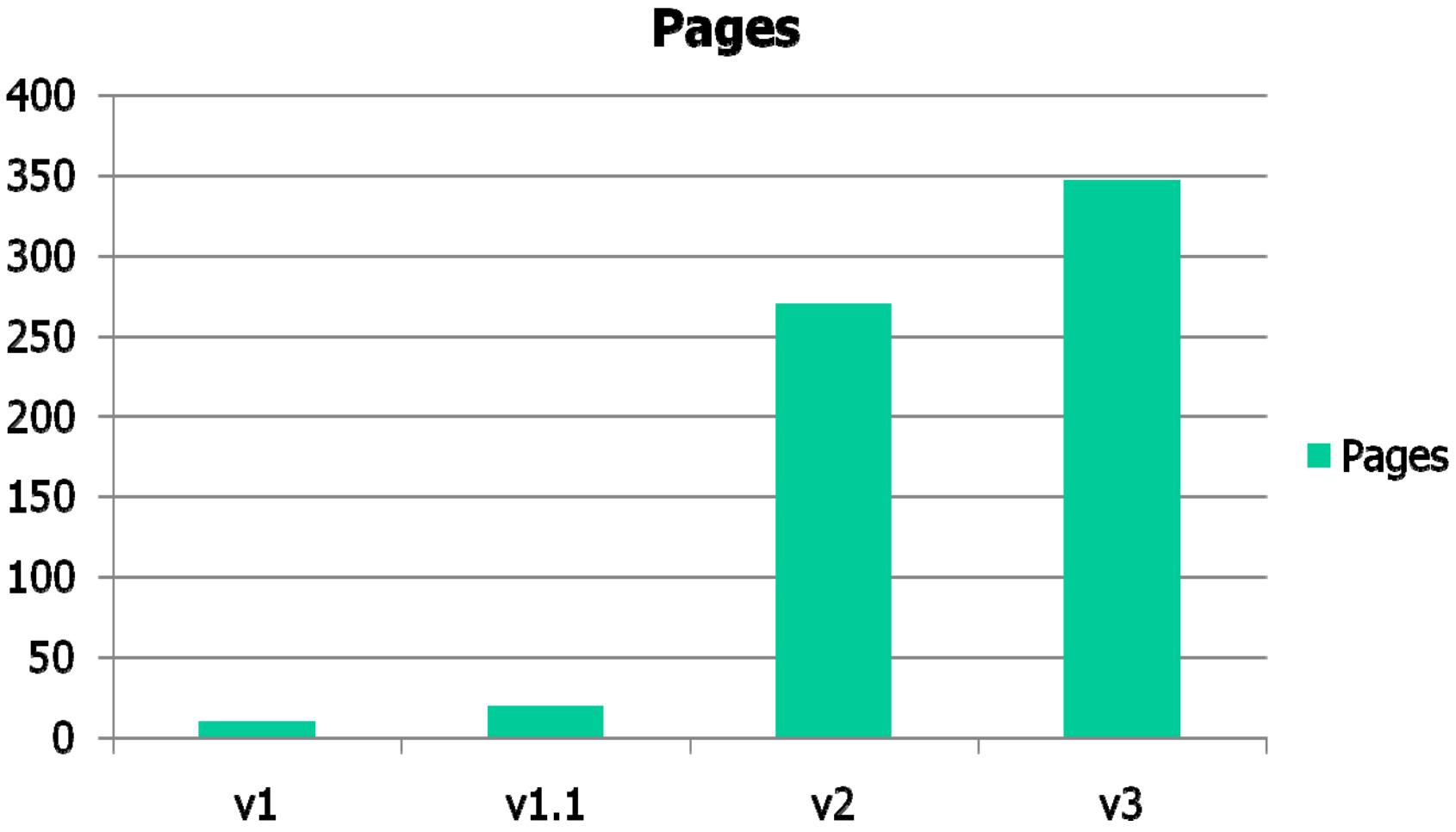


SDLC & OWASP Guidelines in your organization



The OWASP Testing Guide v3

Project History & Complexity



OWASP Testing Guide v3: roadmap

- 26th April 2008: start the new project
- OWASP Leaders brainstorming
- Call for participation: 21 authors
- Index brainstorming
- Discuss the article content
- 20th May 2008: New draft Index
- 1st June 2008: Let's start writing!
- 27th August 2008: started the reviewing phase: 4 Reviewers
  October 2008: Review all the Guide
- December 2008: published the new version of the OWASP Testing Guide: http://www.owasp.org/index.php/OWASP_Testing_Project (347pages +80!)

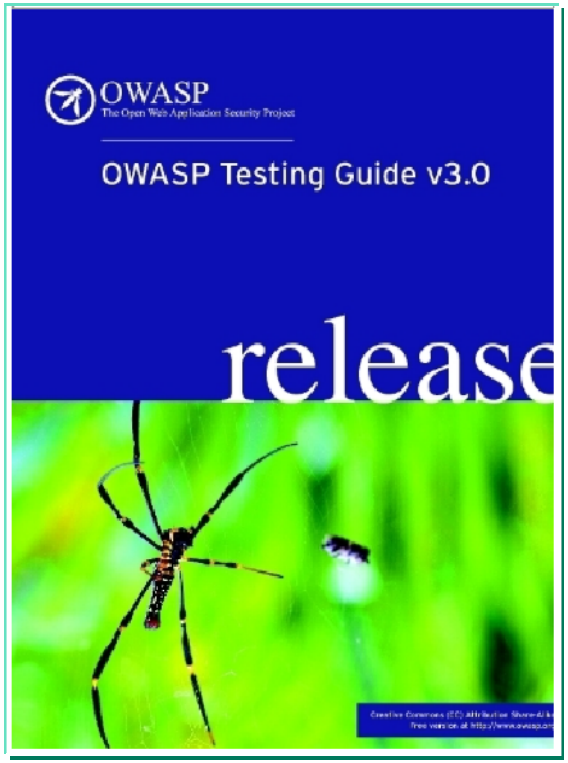
Testing Guide v3: Index

1. Frontispiece
2. Introduction
3. The OWASP Testing Framework
4. Web Application Penetration Testing
5. Writing Reports: value the real risk

Appendix A: Testing Tools

Appendix B: Suggested Reading

Appendix C: Fuzz Vectors



- SANS Top 20 2007
- NIST “Technical Guide to Information Security Testing (Draft)”
- Gary McGraw (CTO Cigital) says: “In my opinion it is the strongest piece of Intellectual Property in the OWASP portfolio”

Testing paragraph template

- **Brief Summary**

  Describe in "natural language" what we want to test. The target of this section is non-technical people (e.g.: client executive)

- **Description of the Issue**

  Short Description of the Issue: Topic and Explanation

- **Black Box testing and example**
  - **How to test for vulnerabilities:**
  - **Result Expected:**

  …
- **Gray Box testing and example**
  - **How to test for vulnerabilities:**
  - **Result Expected:**

  …
- **References**
  - **Whitepapers**
  - **Tools**

Example

Introduction to the methodology

In the next slides we will look at a few examples of tests/attacks and at some real-world cases ....

Information Gathering

- The first phase in security assessment is of course focused on collecting all the information about a target application.
- Using public tools it is possible to force the application to leak information by sending messages that reveal the versions and technologies used by the application
- Available techniques include:
  - Testing: Spiders, robots, and Crawlers (OWASP-IG-001)
  - Search engine discovery/Reconnaissance (OWASP-IG-002)
  - Identify application entry points (OWASP-IG-003)
  - Web Application Fingerprint (OWASP-IG-004)
  - Application Discovery (OWASP-IG-005)
  - Analysis of Error Codes (OWASP-IG-006)

Information Gathering (cont.)

- **Application Fingerprint**

  Knowing the version and type of a running web server allows testers to determine known vulnerabilities and the appropriate exploits to use along the tests. Netcat is the tool of choice for this very well known technique

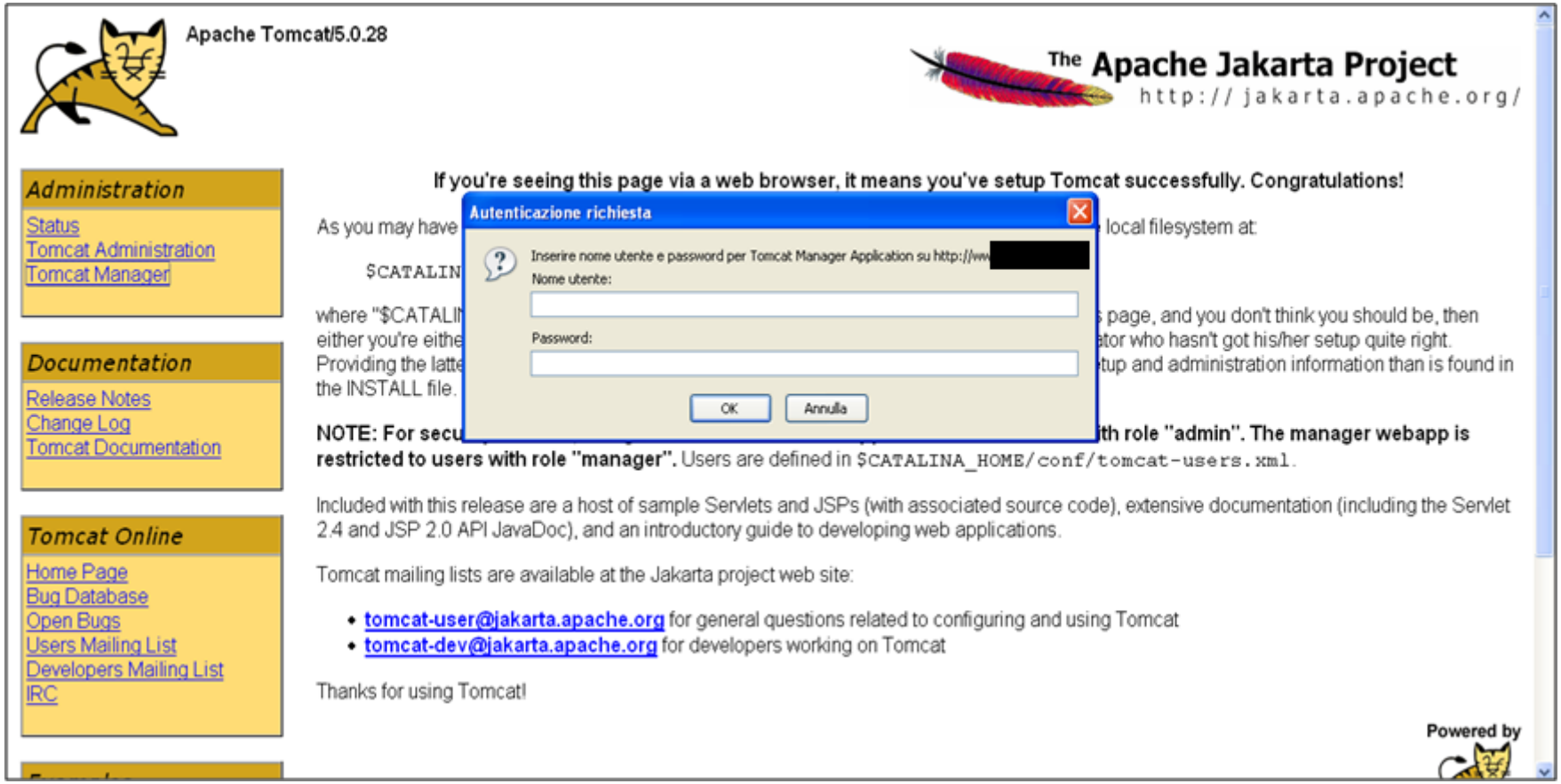
```
$ nc 216.48.3.18 80
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Date: Mon, 16 Jun 2003 02:53:29 GMT
Server: Apache/1.3.3 (Unix)  (Red Hat/Linux)
Last-Modified: Wed, 07 Oct 1998 11:18:14 GMT
ETag: "1813-49b-361b4df6"
Accept-Ranges: bytes
Content-Length: 1179
Connection: close
Content-Type: text/html
```

Configuration Management Testing

- SSL/TLS Testing (OWASP-CM-001)
- DB Listener Testing (OWASP-CM-002)
- Infrastructure Configuration Management Testing (OWASP-CM-003)
- Application Configuration Management Testing (OWASP-CM-004)
- Testing for File Extensions Handling (OWASP-CM-005)
- Old, Backup and Unreferenced Files (OWASP-CM-006)
- Infrastructure and Application Admin Interfaces (OWASP-CM-007)
- Testing for HTTP Methods and XST (OWASP-CM-008)

Configuration Management: example

Access to TomCat Admin Interface



Error codes: example

- http://www.example.com/index.jhtml?ID=2222222222222222222222222222

ERROR:

--- The error occurred in **ibatis**/categorie_abc.xml.

--- The error occurred while preparing the mapped statement for execution.

--- Check the getByLogicalCategoryId.

--- Cause: **java.sql.SQLException:** Invalid parameter object type. Expected 'java.lang.Long' but found 'java.lang.Double'

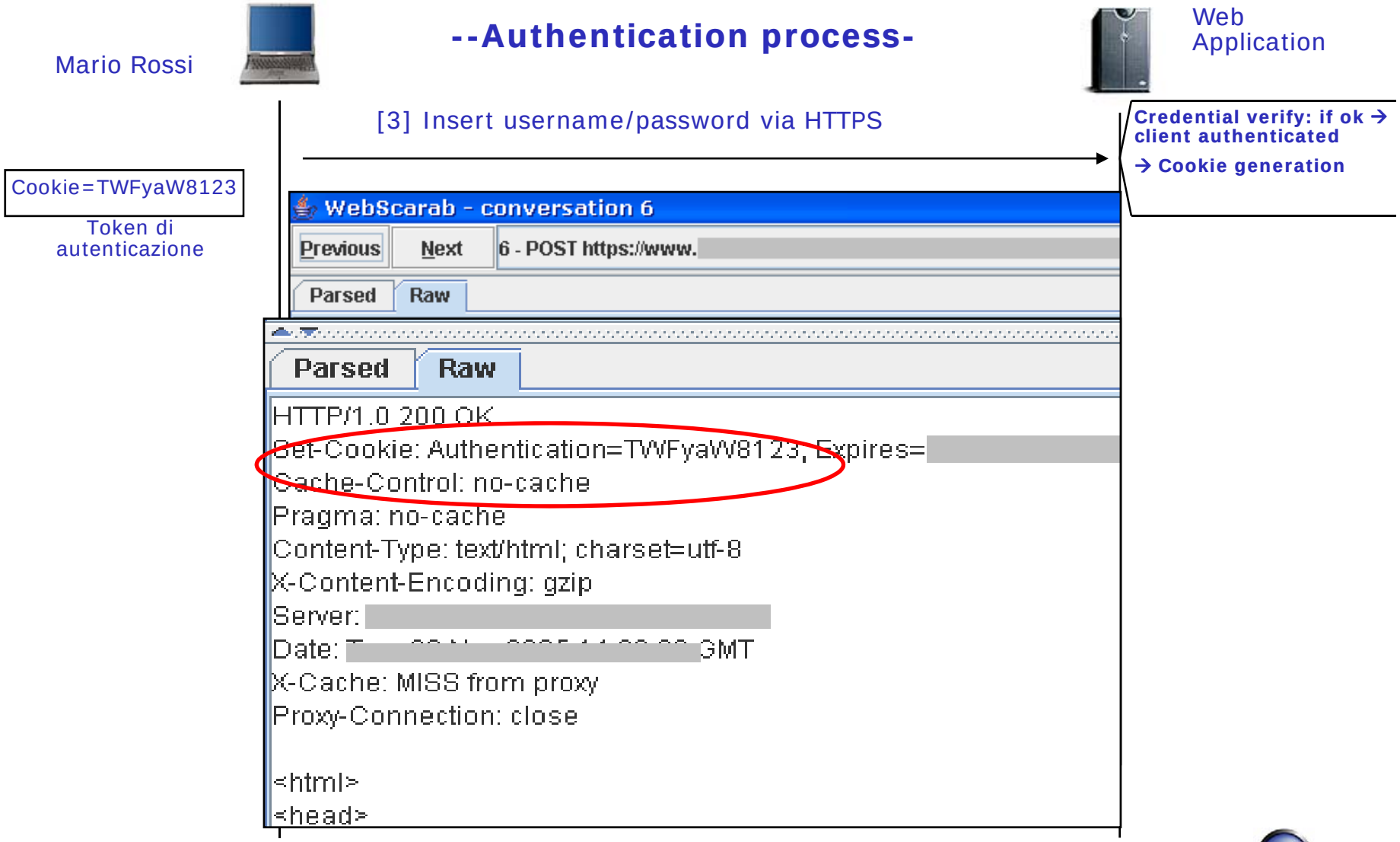
Session management testing

Session management is a critical part of a security test, as every application has to deal with the fact that HTTP is by its nature a stateless protocol. Session Management broadly covers all controls on a user from authentication to leaving the application

Tests include the following areas:

- Testing for session management scheme
- Testing for cookie attributes
- Session Fixation
- Exposed session variables
- Cross Site Request Forgery

Session Management Testing



Session Management Testing

1st Authentication:

User = Mario Rossi; password=12aB45cD:

Cookie=TWFyaW8123

2nd Authentication :

User = Mario Rossi; password=12aB45cD:

Cookie=TWFyaW8125

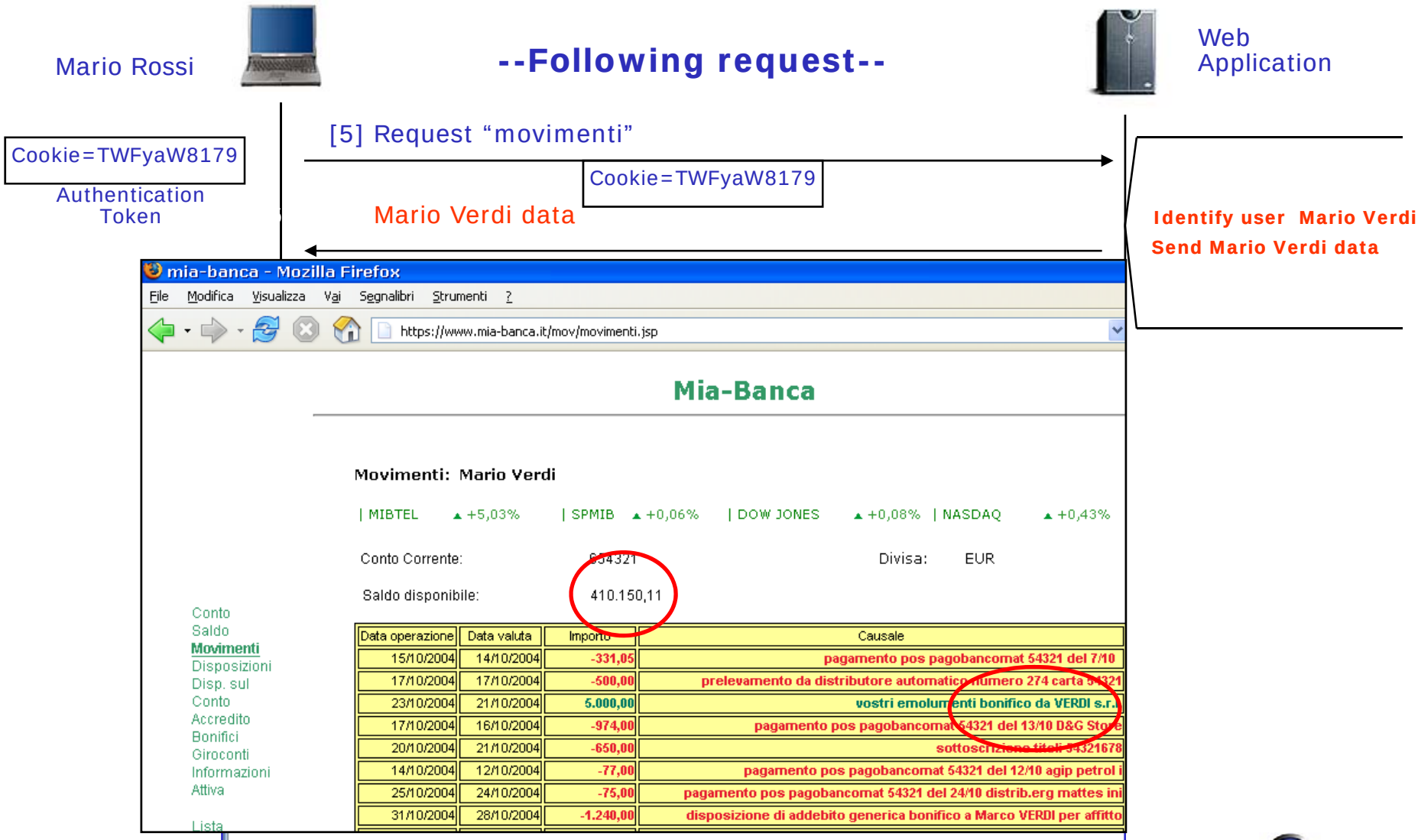
3rd Authentication :

User = Mario Rossi; password=12aB45cD:

Cookie=TWFyaW8127

Cookie Guessable: Cookie=TWFyaW8129

Session Management Testing



Cross Site Request Forgery (CSRF)

- CSRF: When
  - The application permits to send requests in a not authorized manner or send duplicate requests
  - The application uses implicit authentication (session cookie)

- CSRF: How

  The <IMG> attack:
  - A Web site contains an <IMG> TAG inside the HTML code that runs an action on the target site (<IMG> TAG has no restriction to origin level)

Testing for CSRF

- 1st Step: find the vulnerable function
  - Create a new user (admin)
  - Fund transfer (users)
- 2nd Step: Force the user to perform that action
  - Malicious Email
  - Malicious site
- 3rd Step: the user will authenticate on the application
  - Browser will be forced to execute an HTTP request
- Result
  - The authorized action will be executed

Testing for Cross Site Request Forgery

First Step

- Online banking. We analyze the transfer fund mechanism
- We notice that after inserting the receiver coordinate and the money amount we will generate the following HTTP GET
- **https://exampleBank.com/transfer?eu=90000&to=1234”**

Second Step

- User must be authenticated on the application and forced to go to a malicious site or read an email
- IMG tag will execute the request without user interaction

Testing for Cross Site Request Forgery

Third Step

```
<html>
<title> This is a new interesting site..visiting me </title>
<body>
..
<img
src="https://exampleBank.com/transfer?eu=90000&to=1234"
width="0" height="0">
...
</body>
</html>
```

Forth Step

- User browser will execute the action
- There is no way for the application to understand that the action is not forced → log file
- It works!

Authentication testing

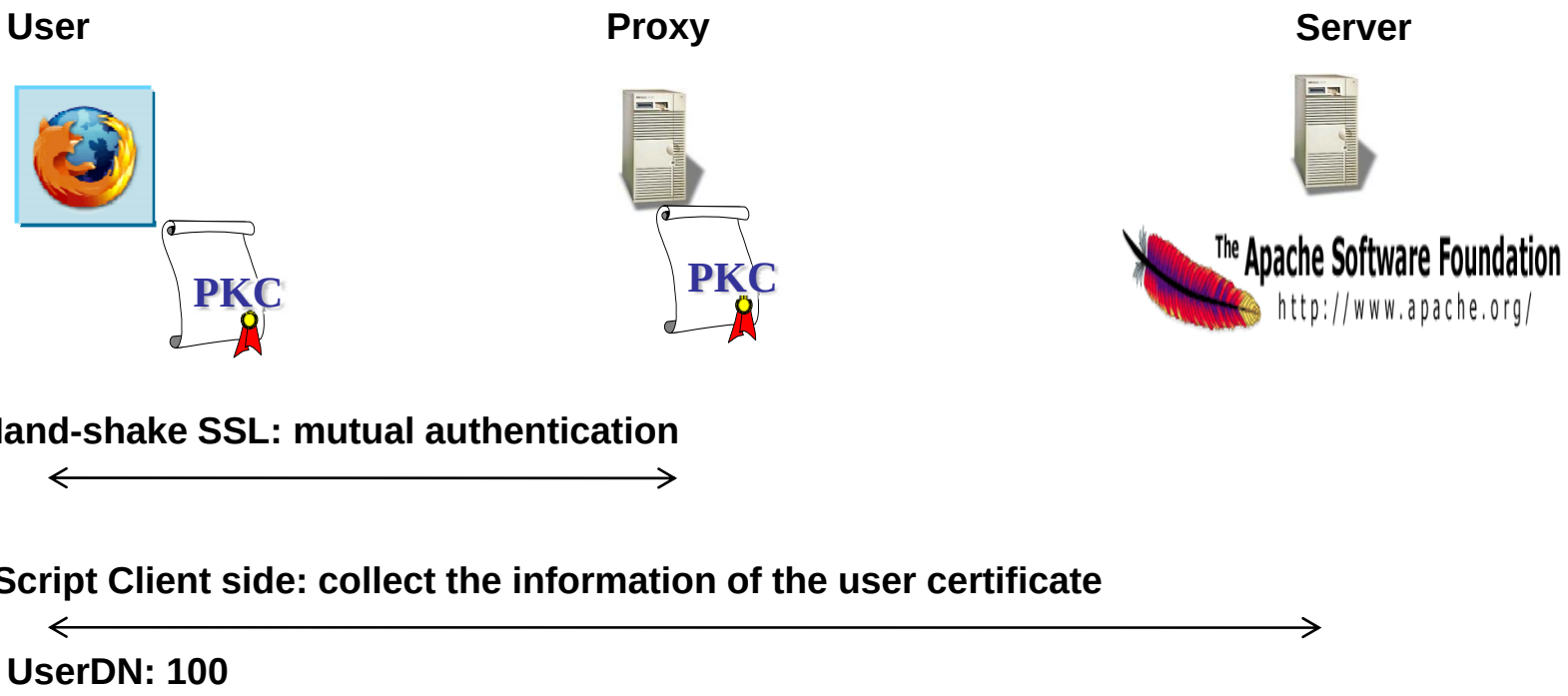
Testing the authentication scheme means understanding how the application checks for users' identity and using that information to circumvent that mechanism and access the application without having the proper credentials

Tests include the following areas:

- Credentials transport over an encrypted channel (OWASP-AT-001)
- Testing for user enumeration (OWASP-AT-002)
- Default or guessable (dictionary) user account (OWASP-AT-003)
- Testing For Brute Force (OWASP-AT-004)
- Testing for Bypassing authentication schema (OWASP-AT-005)
- Testing for Vulnerable remember password and pwd reset (OWASP-AT-006)
- Testing for Logout and Browser Cache Management (OWASP-AT-007)
- Testing for Captcha (OWASP-AT-008)
- Testing for Multiple factors Authentication (OWASP-AT-009)
- Testing for Race Conditions (OWASP-AT-010)

Testing for Bypassing Authentication Schema

- Application with mutual digital certificate authentication:



Testing for Broken Authentication (2)

Here is the Authentication POST:

```
POST https://192.168.1.1:1443/AuthenticationServlet HTTP/1.1
Host: 192.168.1.1:1443
...
Referer: https://192.168.1.1:1443/logonDN.jsp
Cookie:IV_JCT=AncDfj8439Fdfjci454;
Content-Type: application/x-www-form-urlencoded
Content-length: 44

login=true&action=MenuCommand&userDN=100
```

userDN is a value contained in the Digital Certificate . Why the Application does not take this information from the digital certificate received (once verified the CA signature and the certificate integrity)?

Authorization Testing

Authorization is the concept of allowing access to resources only to those permitted to use them. Testing for Authorization means understanding how the authorization process works, and using that information to circumvent the authorization mechanism.

Tests include the following areas:

- Testing for path traversal (OWASP-AZ-001)
- Testing for bypassing authorization schema (OWASP-AZ-002)
- Testing for Privilege Escalation (OWASP-AZ-003)

Testing for privilege escalation

Server Response after the user authentication

```
HTTP/1.1 200 OK
Server: Netscape-Enterprise/6.0
Date: Wed, 1 Apr 2006 13:51:20 GMT
Set-Cookie: USER=aW78ryrGuTWs4MnOd32Fs51yDqp; path=/; domain=.dom.it
Set-Cookie: SESSION=k+KmKpHXTgDi1J5fT7Zz; path=/; domain=.dom.it
Cache-Control: no-cache
Pragma: No-cache
Content-length: 247
Content-Type: text/html
Expires: Thu, 01 Jan 1970 00:00:00 GMT
Connection: close

<form  name="autoriz" method="POST" action = "visual.jsp">
<input type="hidden" name="profile" value="SistInf1">
    <body onload="document.forms.autoriz.submit()">
 </td>
....
```

Authorization parameter

- What if the user modifies the value SistInf1 to SistInf3?

Business logic testing

Business logic may include:

- Business rules that express business policy (such as channels, location, logistics, prices, and products); and
- Workflows based on the ordered tasks of passing documents or data from one participant (a person or a software system) to another.

This step is the most difficult to perform with automated tools, as it requires the penetration tester to perfectly understand the business logic that is (or should be) implemented by the application

Data validation testing

In this phase we test that all input is properly sanitized before being processed by the application, in order to avoid several classes of attacks

- **Cross site scripting (Reflected, Stored, DOM, Flashing)**

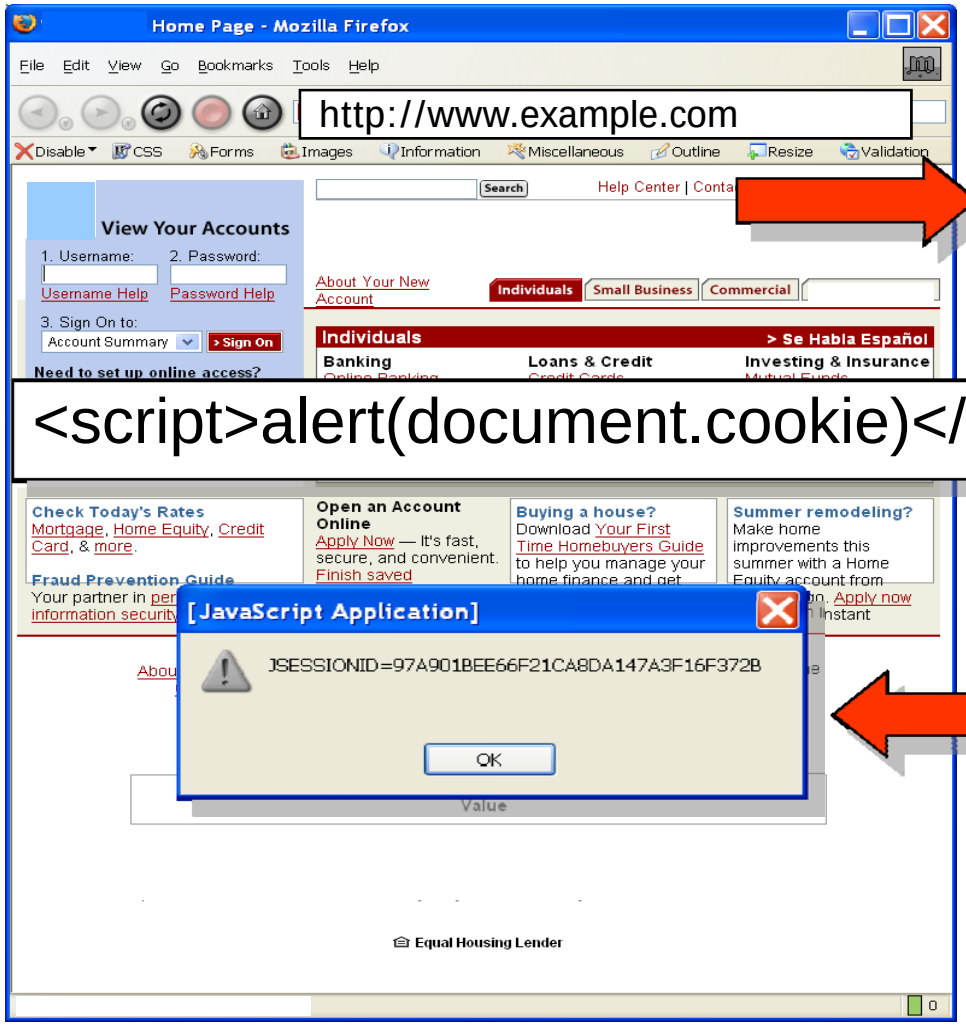
  Test that the application filters JavaScript code that might be executed by the victim in order to steal his/her cookie

- **SQL Injection**

  Test that the application properly filters SQL code embedded in the user input

- **Other attacks based of faulty input validation...**
  - LDAP/XML/SMTP/Command injection
  - Buffer overflows

Reflected Cross Site Scripting



Search field print in output the word searched.

Site sends the script to the user that see his session cookie.

Testing for Command Injection

```
POST http://127.0.0.1:80/WebGoat/attack HTTP/1.1
Host: 127.0.0.1
...
HelpFi
```

```
ExecResults for 'cmd.exe /c type
    "D:\Prog\WebGoat\tomcat\webapps\WebGoat\lesson_plans\"Basic
    Authentication.html | dir c:'
Output...
Il volume nell'unit? C ? WinXP
Numero di serie del volume: 1871-8F02
Directory di C:\
27/12/2007 03.51 0 AUTOEXEC.BAT
27/12/2007 03.51 0 CONFIG.SYS
18/06/2008 09.54 cygwin
18/06/2008 11.43 Dev-Cpp
...
```


Denial of Service Testing

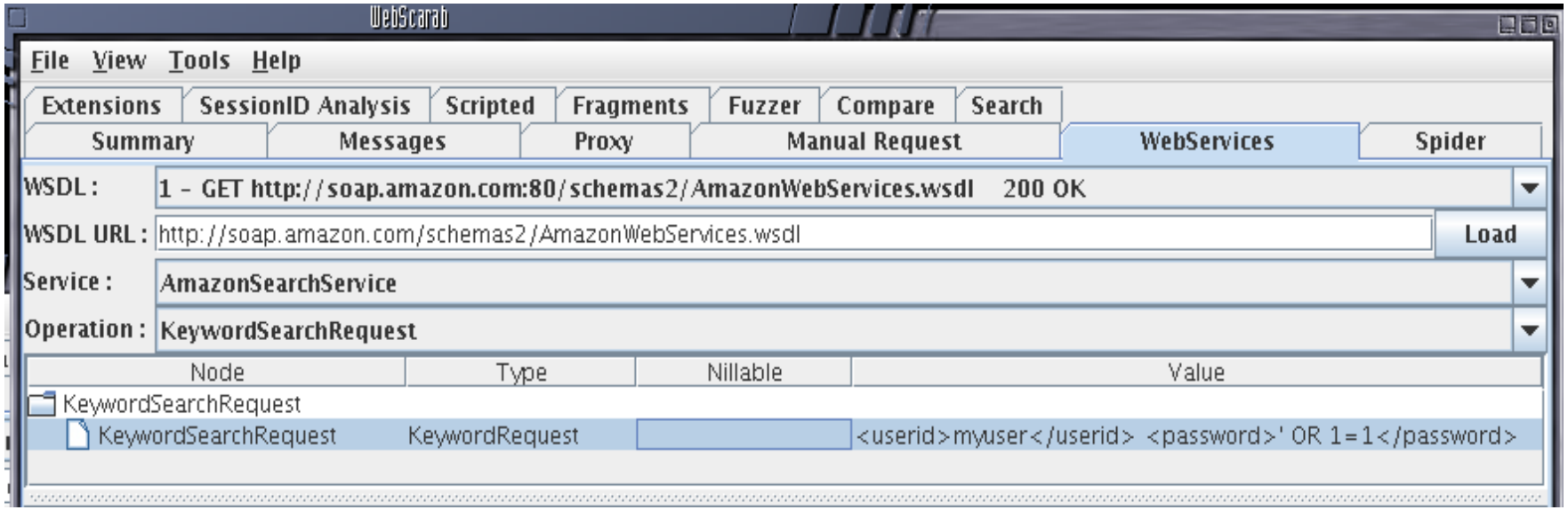
DoS are types of vulnerabilities within applications that can allow a malicious user to make certain functionality or sometimes the entire website unavailable. These problems are caused by bugs in the application, often resulting from malicious or unexpected user input

- Testing for SQL Wildcard Attacks (OWASP-DS-001)
- Locking Customer Accounts (OWASP-DS-002)
- Buffer Overflows (OWASP-DS-003)
- User Specified Object Allocation (OWASP-DS-004)
- User Input as a Loop Counter (OWASP-DS-005)
- Writing User Provided Data to Disk (OWASP-DS-006)
- Failure to Release Resources (OWASP-DS-007)
- Storing too Much Data in Session (OWASP-DS-008)

Usually not performed in performed on production environments

Web Services Testing

- The vulnerabilities are similar to other “classical” vulnerabilities such as SQL injection, information disclosure and leakage etc but web services also have unique XML/parser related vulnerabilities.
- WebScarab (available for free at www.owasp.org) provides a plug-in specifically targeted to Web Services. It can be used to craft SOAP messages that contains malicious elements in order to test how the remote system validates input



Web Services Testing

XML Structural Testing

In this example, we see a snippet of XML code that violates the hierarchical structure of this language. A Web Service must be able to handle this kind of exceptions in a secure way

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<note id="666">
<to>OWASP
<from>EOIN</from>
<heading>I am Malformed </to>
</heading>
<body>Example of XML Structural Test</body>
</note>
```

Web Services Testing (cont.)

XML Large payload

Another possible attack consists in sending to a Web Service a very large payload in an XML message. Such a message might deplete the resource of a DOM parser

```
<Envelope>
<Header>
   <wsse:Security>
     <Hehehe>I am a Large String (1MB)</Hehehe>
     <Hehehe>I am a Large String (1MB)</Hehehe>
     <Hehehe>I am a Large String (1MB)</Hehehe>…
    <Signature>…</Signature>
   </wsse:Security>
 </Header>
 <Body>
   <BuyCopy><ISBN>0098666891726</ISBN></BuyCopy>
 </Body></Envelope>
```

Testing Report: model

- The OWASP Risk Rating Methodology
  - Estimate the severity of all of these risks to your business
  - This is not universal risk rating system: vulnerability that is critical to one organization may not be very important to another

- Simple approach to be tailored for every case
  - standard risk model: **Risk = Likelihood * Impact**

- Identifying a risk

  You'll need to gather information about:
  - the vulnerability involved
  - the threat agent involved
  - the attack they're using
  - the impact of a successful exploit on your business.

Writing Report

- I. Executive Summary
- II. Technical Management Overview
- III Assessment Findings
- IV Toolbox

Category	Ref. Number	Name	Affected Item	Finding	Comment/Solution	Risk
Authentication Testing	OWASP-AT-003	Bypassing authentication schema				
	OWASP-AT-004	Directory traversal/file include				
	OWASP-AT-005	Vulnerable remember password and pwd reset				
	OWASP-AT-006	Logout and Browser Cache Management Testing				
Session Management	OWASP-SM-001	Session Management Schema				
	OWASP-	Session Token				

How the Guide will help the security industry

Pen-testers

- A structured approach to the testing activities
- A checklist to be followed
- A learning and training tool

Clients

- A tool to understand web vulnerabilities and their impact
- A way to check the quality of the penetration tests they buy

More in general, the Guide aims to provide a pen-testing standard that creates a 'common ground' between the pen-testing industry and its client.

This will raise the overall quality and understanding of this kind of activity and therefore the general level of security in our infrastructures

Thanks!

V3 Authors

- Anurag Agarwwal
- Daniele Bellucci
- Arian Coronel
- Stefano Di Paola
- Giorgio Fedon
- Alan Goodman
- Christian Heinrich
- Kevin Horvath
- Gianrico Ingrosso
- Roberto Suggi Liverani
- Alex Kuza
- Pavol Luptak
- Ferruh Mavituna
- Marco Mella
- Matteo Meucci
- Marco Morana
- Antonio Parata
- Cecil Su
- Harish Skanda Sureddy
- Mark Roxberry
- Andrew Van der Stock

V3 Reviewers

- Marco Cova
- Nam Nguyen
- Kevin Fuller
- Matteo Meucci

References

OWASP Italy:
www.owasp.org/index.php/Italy

OWASP Italy mailing list:
http://lists.owasp.org/mailman/listinfo/owasp-italy

OWASP Testing Guide:
http://www.owasp.org/index.php/Category:OWASP_Testing_Project

Gary McGraw OWASP PodCast:
http://www.owasp.org/index.php/Podcast_5

Thank you!

http://www.owasp.org

matteo.meucci@owasp.org