OWASP e gli standard per la sicurezza applicativa

Matteo Meucci

OWASP-Italy Chair

OWASP Day per la PA
Roma
5, Novembre 2009

MEF
Ministero dell'Economia e delle Finanze

The OWASP Foundation
http://www.owasp.org

Agenda

- Introduzione alla Web Application Security
- Il progetto OWASP (The Open Web Application Security Project)
- Quali strumenti per implementare software sicuro e difendersi da possibili minacce

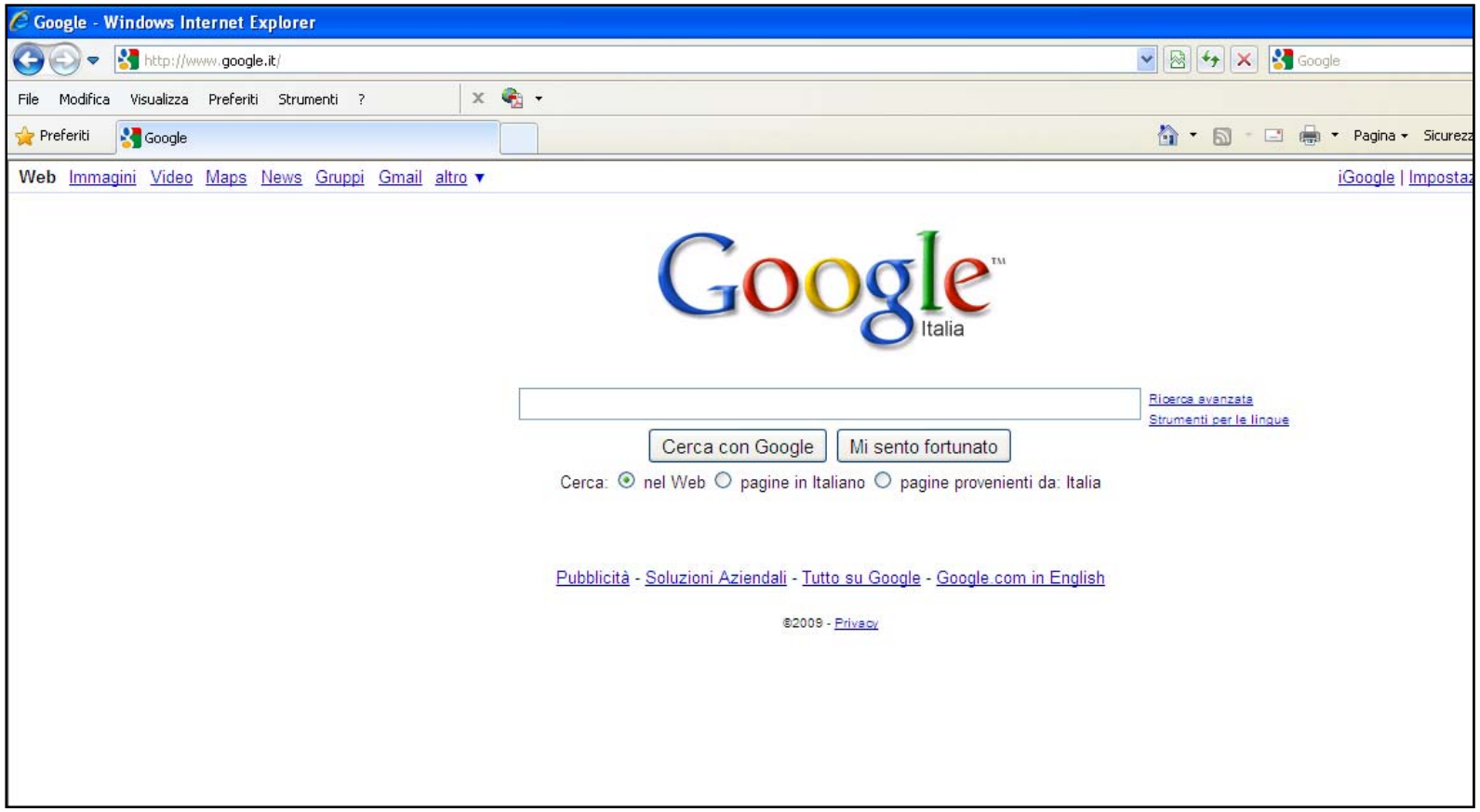
Who am I?

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

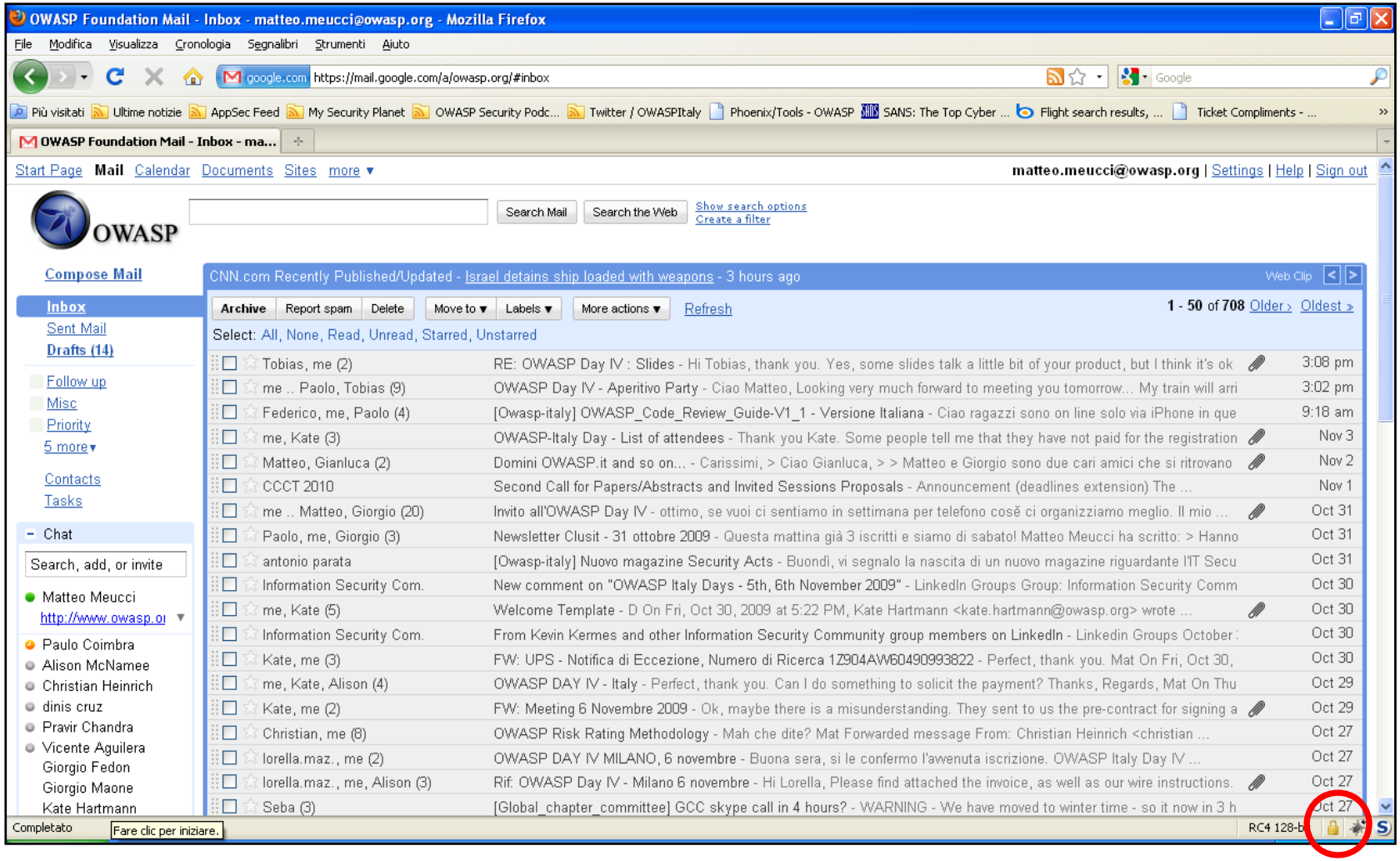
- Work
  - CEO @ Minded Security
    Application Security Consulting
  - 8+ years on Information Security
    focusing on Application Security

Introduzione alla Web Application Security

Focus: applicazioni, software



Applicativo sicuro o insicuro?



Ingredienti del software sicuro?

Ingredienti: Sun Java 1.5 runtime, Sun J2EE 1.2.2, Jakarta log4j 1.5, Jakarta Commons 2.1, Jakarta Struts 2.0, Harold XOM 1.1rc4, Hunter JDOMv1

Software Facts

Expected Number of Users 15
Typical Roles per Instance 4

Amount Per Serving

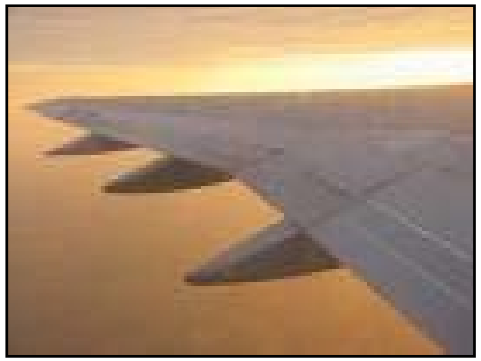
Modules 155 Modules from Libraries 120

	% Vulnerability*
Cross Site Scripting 22	65%
Reflected 12	15%
Stored 10	
SQL Injection 2	10%
Buffer Overflow 5	95%
Total Security Mechanisms 3	10%
Modularity .035	0%
Cyclomatic Complexity 323	
Encryption 3	
Authentication 15	4%
Access Control 3	2%
Input Validation 233	20%
Logging 33	4%

* % Vulnerability values are based on typical use scenarios for this product. Your Vulnerability Values may be higher or lower depending on your software security needs:

	Usage	Intranet	Internet
Cross Site Scripting	Less Than	10	5
Reflected	Less Than	10	5
Stored	Less Than	10	5
SQL Injection	Less Than	20	2
Buffer Overflow	Less Than	20	2
Security Mechanisms		10	14
Encryption		3	15

La verifica di sicurezza del software



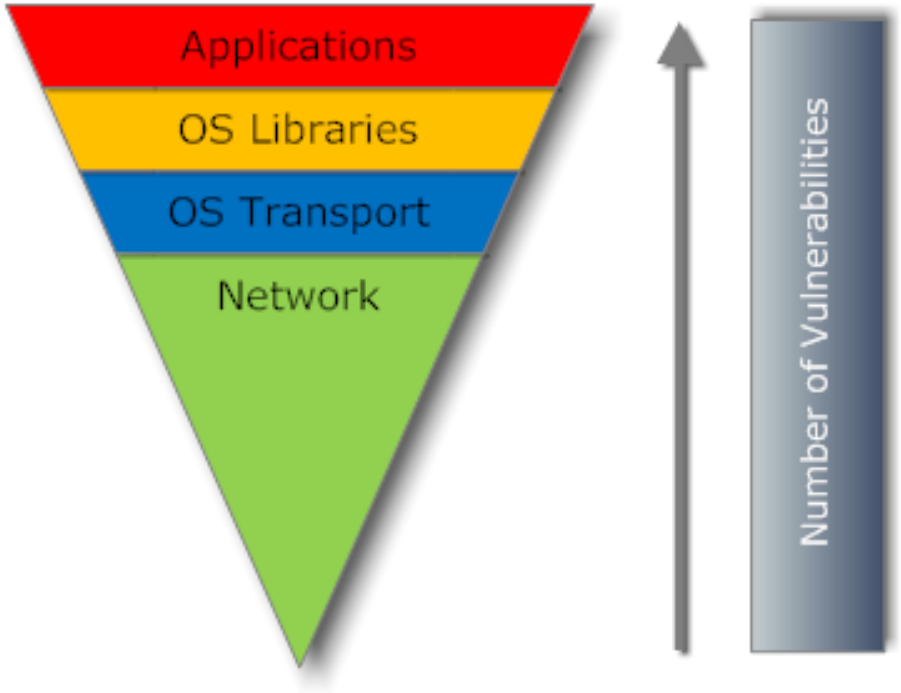
```
public class HelloWorld extends HttpServlet {

    public void doGet(
            HttpServletRequest request,
            HttpServletResponse response)
                throws IOException, ServletException
    {
            response.setContentType("text/html");
            PrintWriter out = response.getWriter();
            out.println("<HTML><HEAD>");
            out.println("<TITLE>Hello World</TITLE>");
            out.println("</HEAD><BODY>");
            out.println("Hello, " + }
```

?

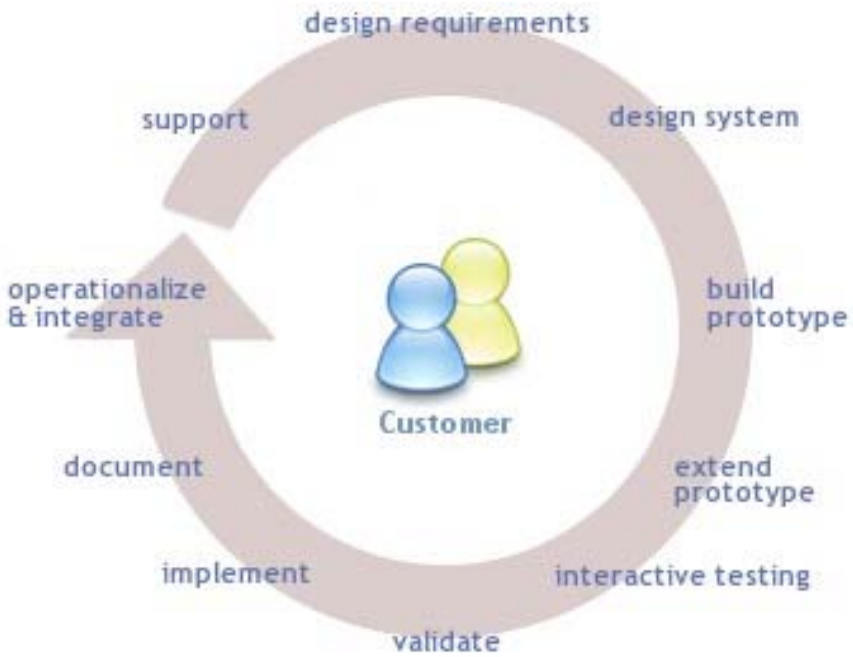
Il controllo dei difetti di sicurezza del software dovrebbe essere considerato parte del processo di sviluppo del software.

Dove risiedono le maggiori vulnerabilità?



Source SANS : The Top Cyber Security Risks (Set 09)

Perchè le applicazioni web rappresentano il maggior problema di sicurezza oggi?



Ciclo di vita di un'applicazione

Source: www.linuxbox.com

- Time-to-Market
  - Le applicazioni devono essere sul disponibili il prima possibile
- Complessità crescente
  - Il ciclo di vita delle applicazioni ha complessità sempre più crescente
- Crescente domanda di business
  - Funzionalità vs Sicurezza
- → Minor priorità alle funzioni ed alle caratteristiche di sicurezza

Web Application Security

- La sicurezza applicativa comprende tutti i processi che introducono i controlli di sicurezza durante il ciclo di vita di sviluppo del software.
- Per garantire la protezione dell'applicazione Web, è necessario considerare le **vulnerabilità come difetti**. Di conseguenza, la protezione delle applicazioni deve rappresentare una pratica integrata nei processi di gestione della qualità durante il ciclo di sviluppo delle applicazioni.
- Si parla di Web Application Security quando un'azienda:

  progetta, sviluppa e testa i propri applicativi con processi consolidati ed utilizzando linee guida e standard di riferimento (OWASP).

Vulnerabilità possibili delle applicazioni web

- Information Disclosure
- SSL Weakness
- Configuration management weakness
- Old, backup and unreferenced files
- Access to Admin interfaces
- HTTP Methods enabled, XST permitted, HTTP Verb
- Credentials transport over an encrypted channel
- User enumeration
- Guessable user account
- Credentials Brute forcing
- Bypassing authentication schema
- Vulnerable remember pwd weak pwd reset
- Logout function
- browser cache weakness
- Bypassing Session Management Schema, Weak Session Token
- Cookies not secure
- Session Fixation
- Exposed sensitive session variables
- CSRF
- Path Traversal
- Bypassing authorization schema
- Privilege Escalation
- Bypassable business logic
- Reflected XSS,Stored XSS,DOM XSS
- Cross Site Flashing
- SQL, LDAP. ORM, XML,SSI, Code Injection
- OS Commanding
- Buffer overflow
- Locking Customer Accounts
- Buffer Overflows
- WSDL Weakness

Minacce

- La mancanza di policy nella scelta delle password può portare all'individuazione di username/password di un insieme di clienti
- Un meccanismo debole di autenticazione può permettere il bypass dello schema di autenticazione (furto di identità)
- Un meccanismo di autorizzazione debole può risultare nella individuazione di informazioni riservate, o la possibilità di accedere a funzionalità non autorizzate
- Furto della sessione temporanea dell'utente (controllo temporaneo dell'accesso all'applicazione)
- Forzare un utente ad eseguire un'azione non voluta (es. bonifico)
- Attacchi sul browser dei clienti (furto di identità e di informazioni riservate degli utenti)
- Controllo dei server ospitanti l'applicazione e database
- Intercettazione delle informazioni in transito dall'utente al server e di username/password

Gli impatti delle vulnerabilità:

In generale le vulnerabilità applicative portano a:

- Perdita/manipolazione di Dati
- Manipolazione della presentazione delle informazioni
- Perdita di fiducia, di immagine
- **Perdita di Clienti**

Esempi:

- Applicazione compromessa in cui vengono installate applicazioni
  (es: malware, repository di file illeciti, redirect su siti illeciti)
- Disclosure: le vulnerabilità sono pubblicate su paper/siti

Strategie di difesa

Come può una PA difendersi e gestire tutte le problematiche di sviluppo sicuro?

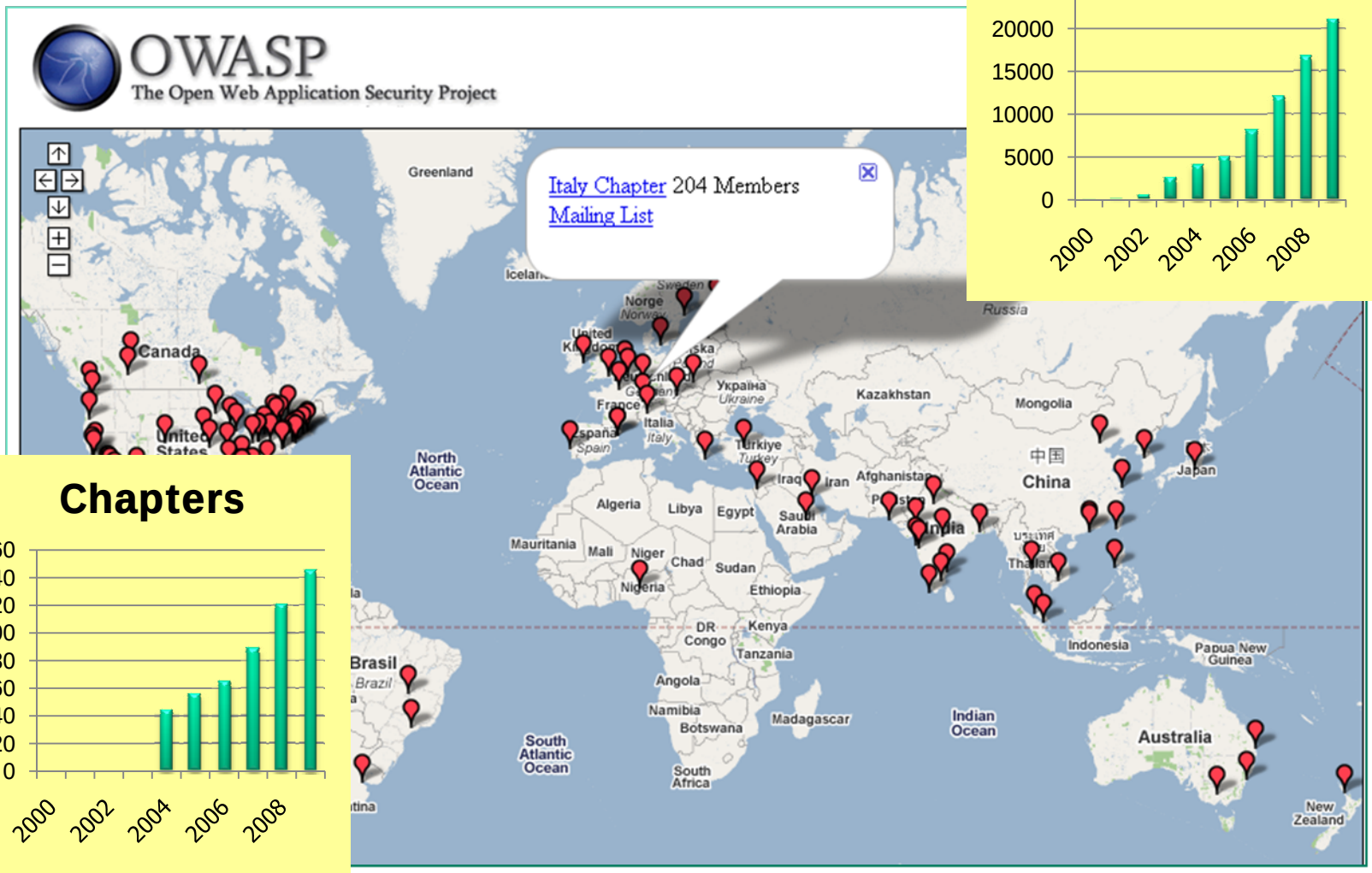
- Cultura, formazione continua
- Adottare linee guida di sviluppo sicuro
- Creare processi di:
  - review del codice
  - verifica dell'applicazione
- Monitorare il proprio processo di sviluppo sicuro

OWASP: The Open Web Application Security Project

OWASP
The Open Web Application Security Project

- Il progetto Open Web Application Security Project (OWASP) è una organizzazione Open Source dedicata alla creazione e alla diffusione di una cultura per quanto riguarda la sicurezza delle applicazioni web
- Progetto free, come il materiale disponibile sul portale www.owasp.org
- Migliaia di membri, +100 capitoli locali e altri partecipanti ai progetti. Milioni di hit su www.owasp.org al mese
- Defense Information Systems Agency (DISA) , US Federal Trade Commission (FTC), VISA, Mastercard, American Express e molte aziende in Italia hanno adottato la documentazione OWASP nei loro standard e linee guida

OWASP Worldwide Community

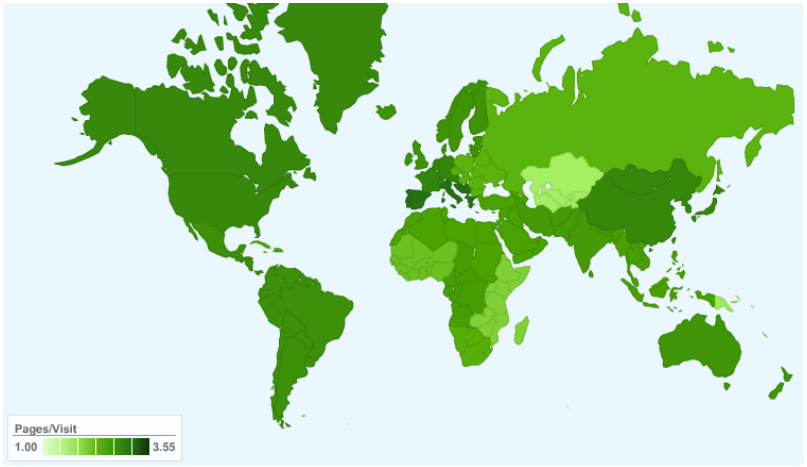


Participants

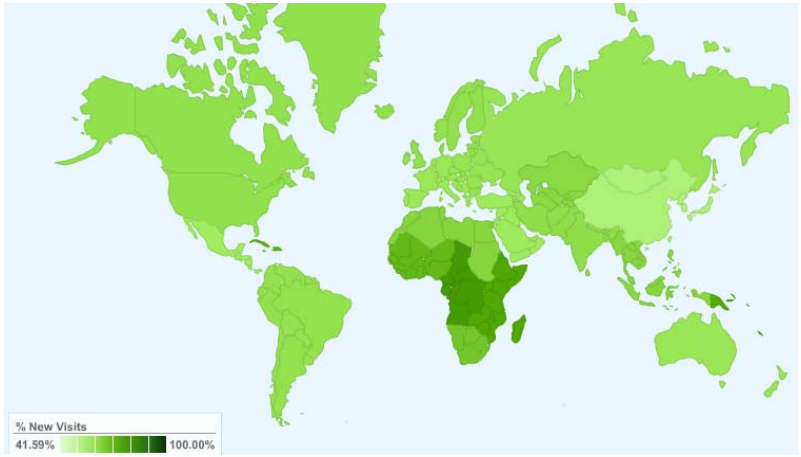
Chapters

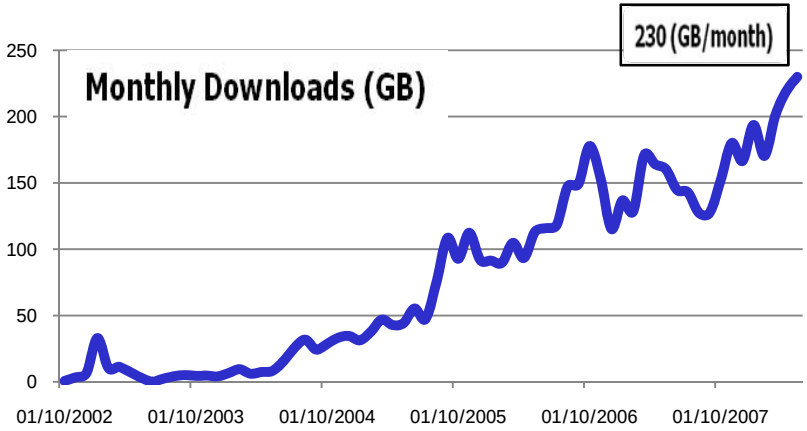
OWASP Dashboard

Worldwide Users

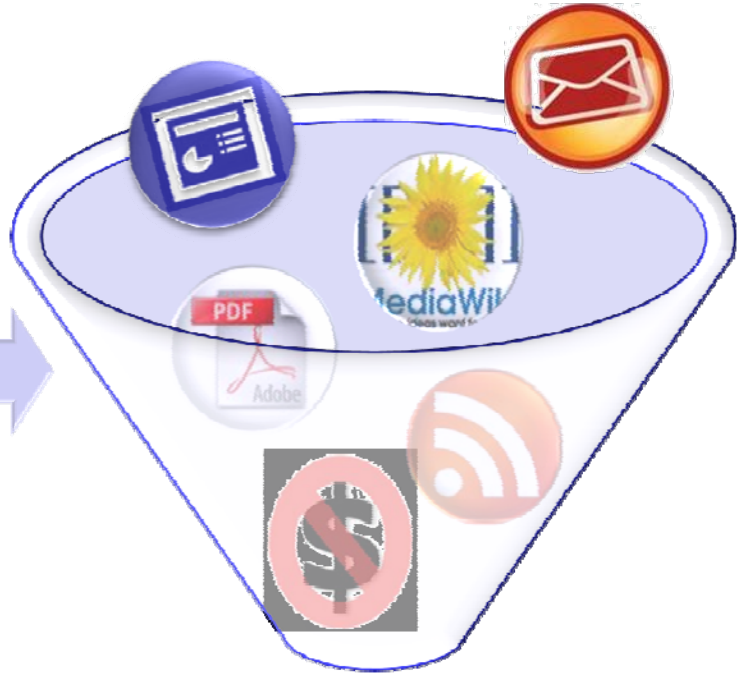


Most New Visitors





La base di conoscenza di OWASP



- 6,381 Articoli
- 427 presentazioni
- 200 aggiornamenti/giorno
- 271 mailing lists
- 180 blog monitorati

OWASP Top Ten

www.owasp.org/index.php?title=Top_10_2007

- A1: Cross Site Scripting (XSS)
- A2: Injection Flaws
- A3: Malicious File Execution
- A4: Insecure Direct Object Reference
- A5: Cross Site Request Forgery (CSRF)
- A6: Information Leakage and Improper Error Handling
- A7: Broken Authentication and Session Management
- A8: Insecure Cryptographic Storage
- A9: Insecure Communications
- A10: Failure to Restrict URL Access

OWASP
The Open Web Application Security Project
http://www.owasp.org

PCI Security Standards Council

Linee Guida OWASP

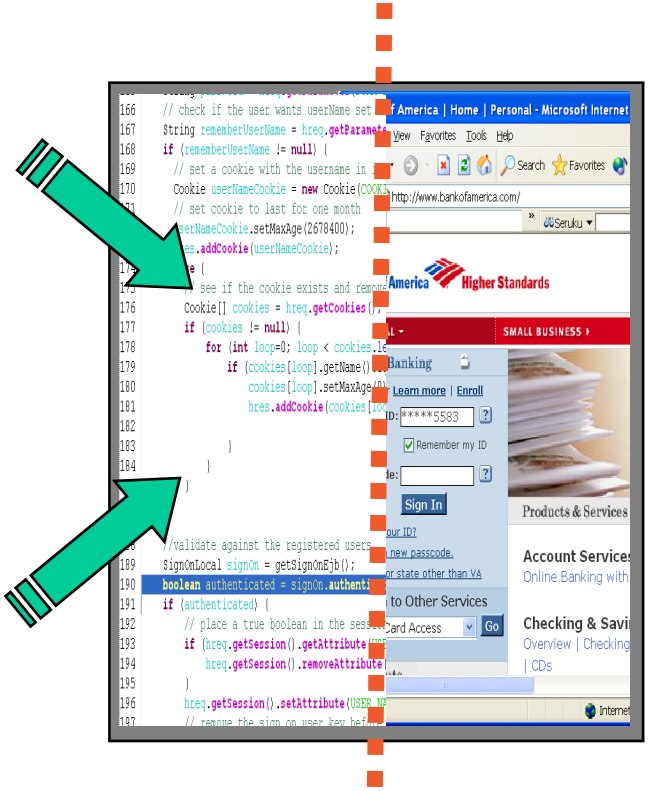
- Gratuite e open source
- Libri a basso costo
- Coprono tutti i controlli di sicurezza
- Centinaia di esperti
- Tutti gli aspetti di sicurezza applicativa

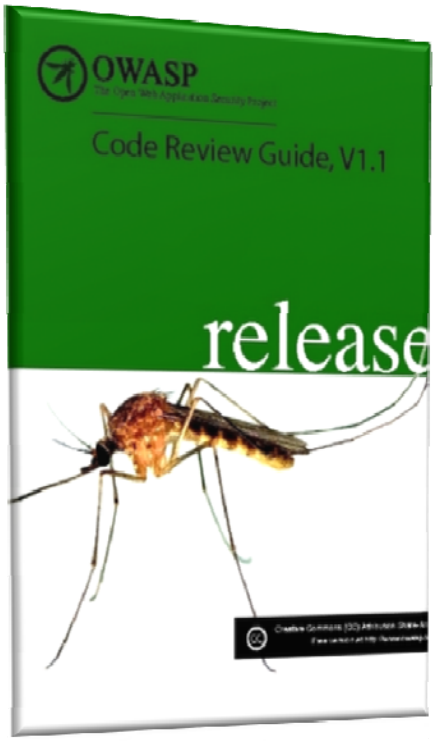
OWASP Building Guide

- Al fine di comprendere ed eliminare le cause della “insicurezza” nel software,OWASP ha sviluppato la guida per lo sviluppo delle applicazioni web sicure pensata per:
  - Sviluppatori per implementare i meccanismi di sicurezza ed evitare le vulnerabilità;
  - Project manager che la utilizzano per identificare le attività da svolgere (threat modeling, code review, development);
  - Team di sicurezza che la usano per apprendere le tematiche di application security e l’approccio per la messa in sicurezza;

OWASP Code Review Guide

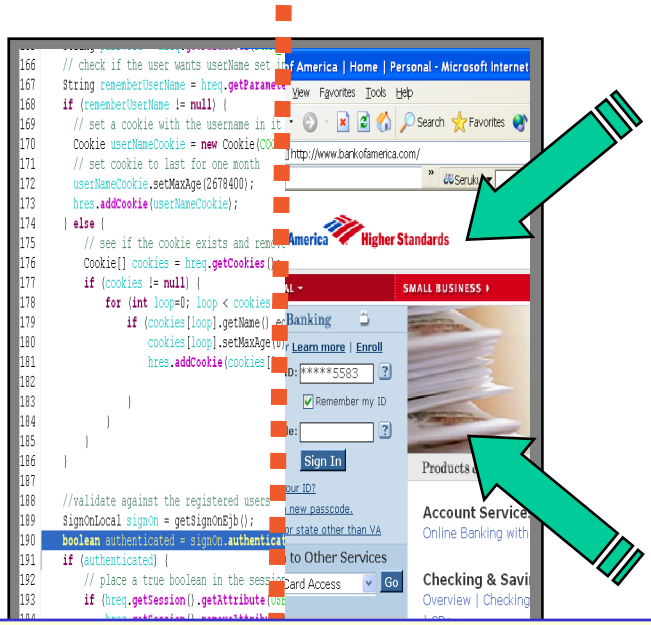
- Descrive la metodologia OWASP per testare il codice di un'applicazione (white box testing, conoscendo il codice sorgente)





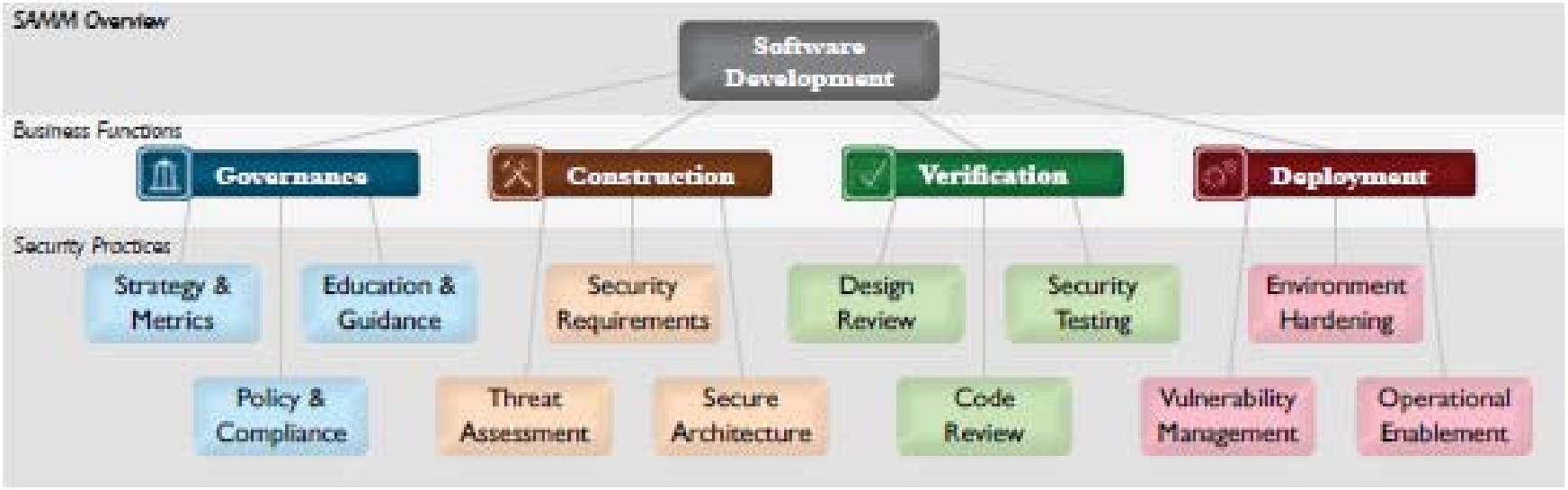
OWASP Testing Guide

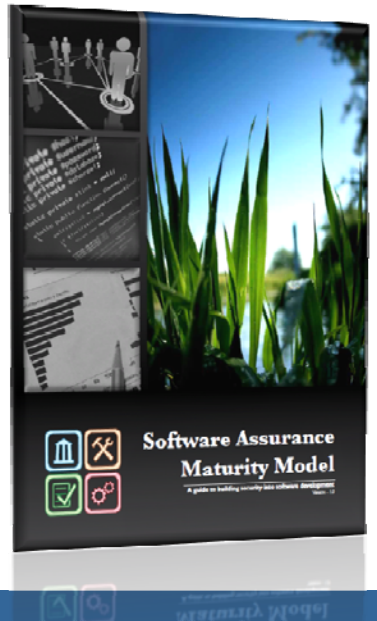
- Descrive la metodologia OWASP per testare la sicurezza di un applicativo web



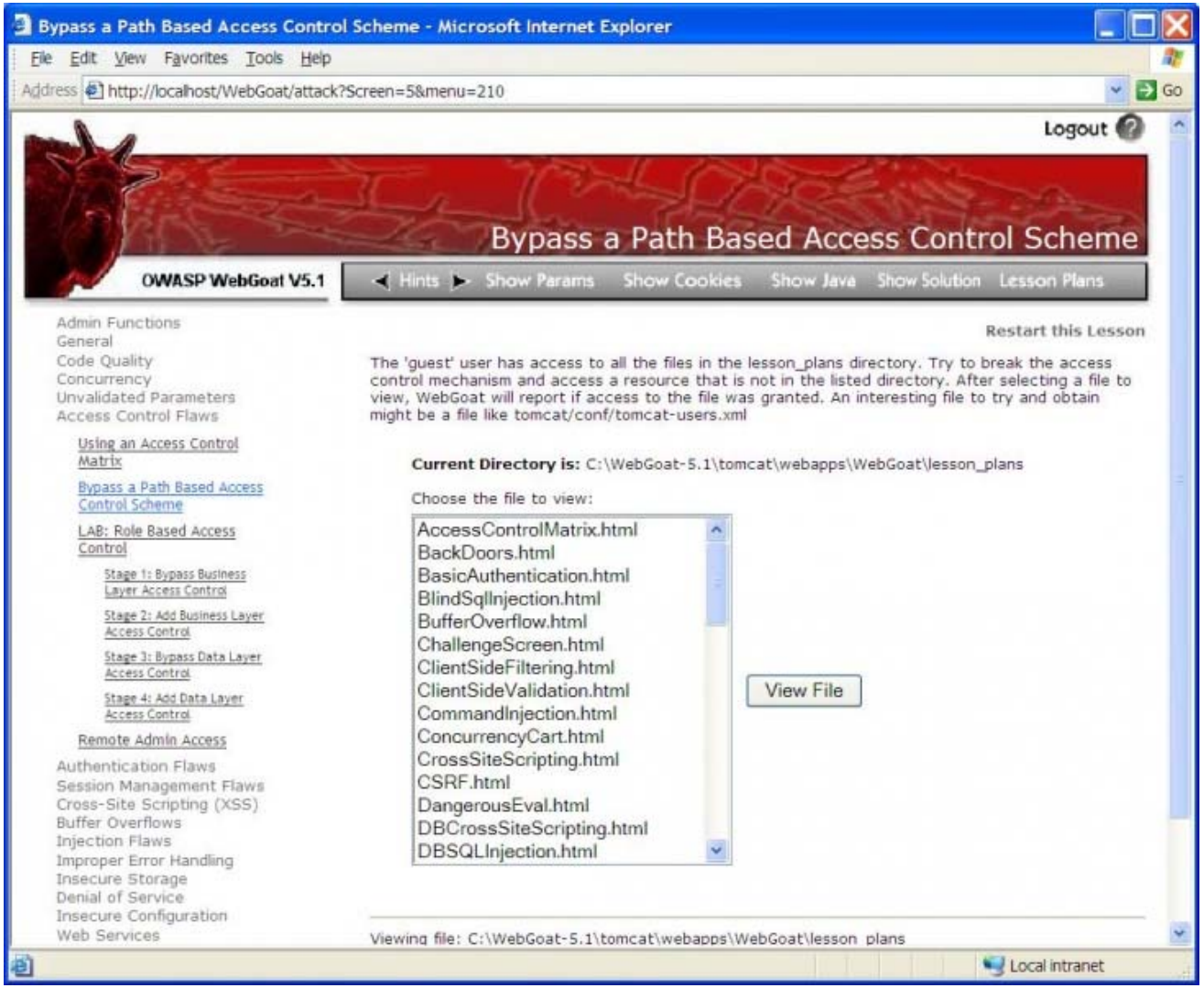
- **SANS Top 20 2007**
- **NIST "Technical Guide to Information Security Testing (Draft)"**
- **Gary McGraw (CTO Cigital) says: "In my opinion it is the strongest piece of Intellectual Property in the OWASP portfolio"**

OWASP Software Assurance Maturity Model

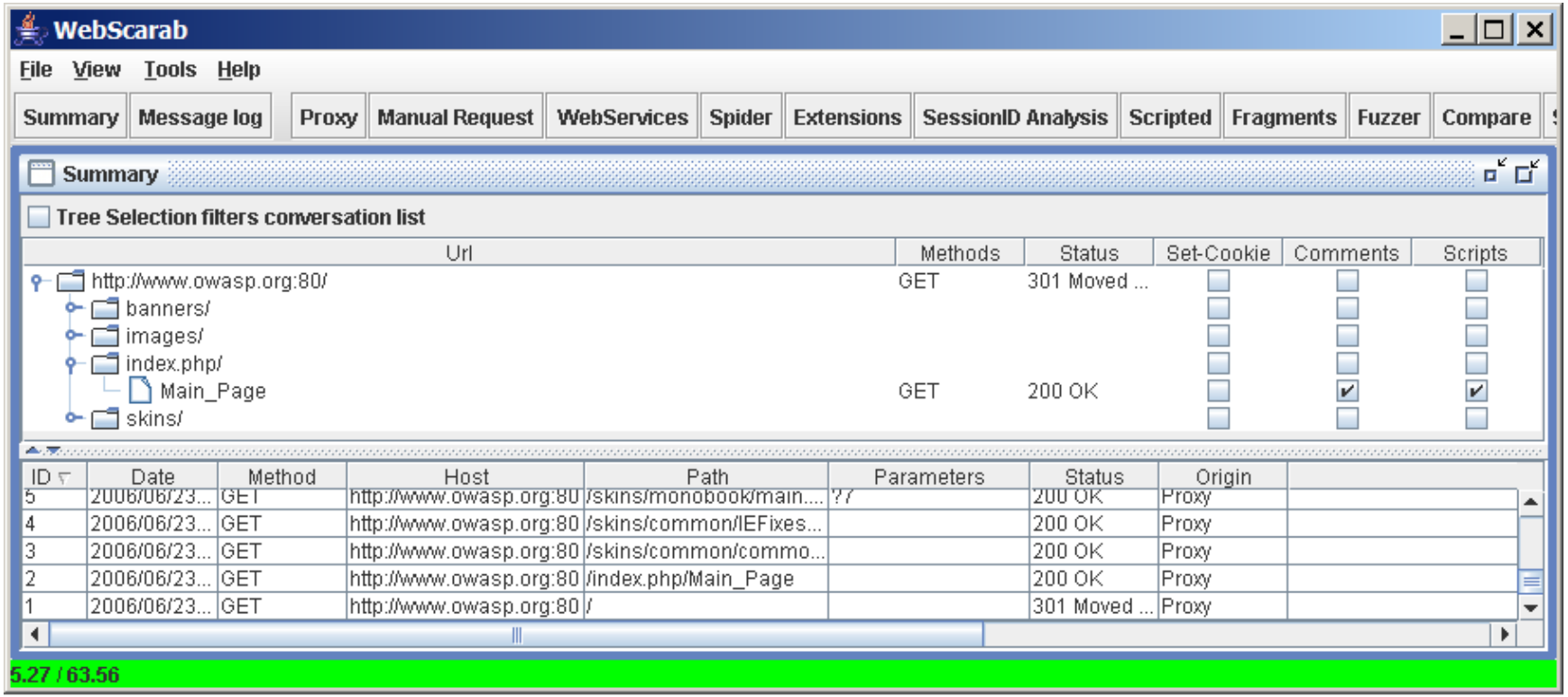




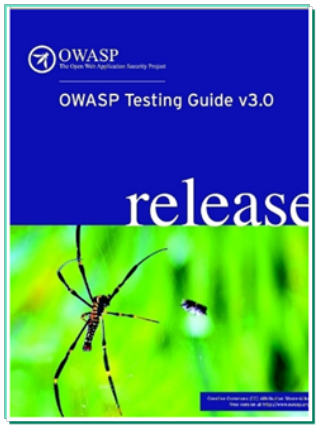
OWASP WebGoat



OWASP WebScarab

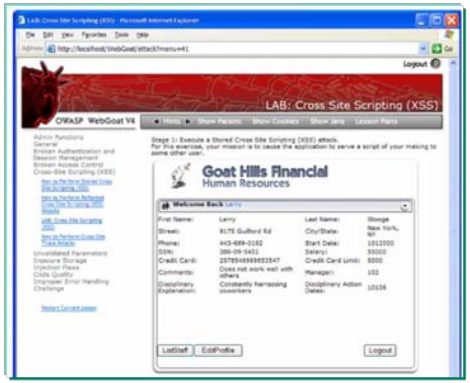


Principali progetti OWASP



BOOKS

- Owasp top10
- Building guide
- Code review guide
- Testing guide



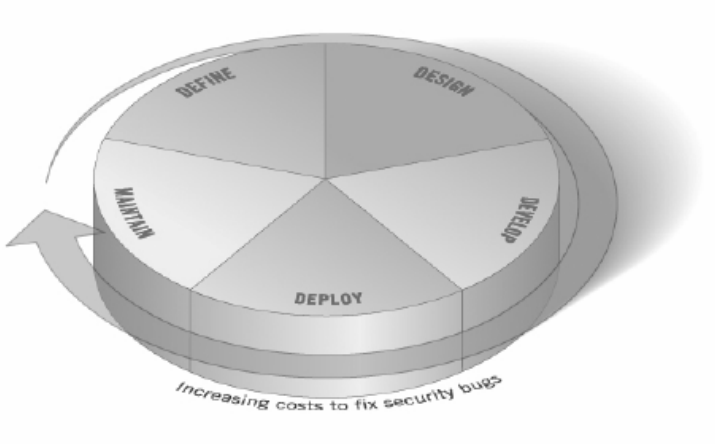
TOOLS

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

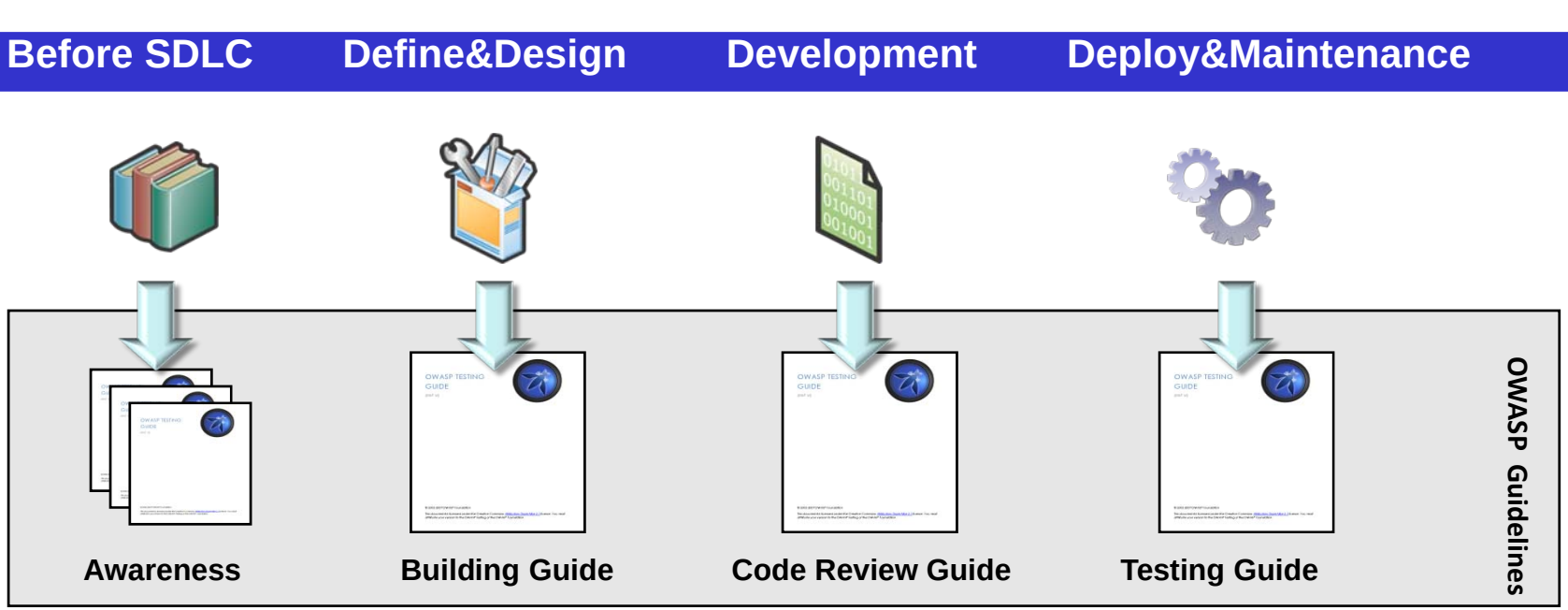
Il ciclo di vita del software e la sicurezza

Il ciclo di vita del software

- Il Ciclo di Vita del Software (Sofware Development Life Cycle, SDLC) comprende :
  - Define
  - Design
  - Develop
  - Deploy
  - Maintain
- Quali processi implementare?
  - Awareness
  - Secure Code Guidelines
  - Code Review
  - Application Testing



SDLC & OWASP Guidelines e tools



Verifica della sicurezza

- In-house o terza parte?
- Code Review o Application Testing?
- Adozione di tool o analisi manuale?

Verifica della sicurezza: in-house

- Vantaggi:
  - Portare cultura in azienda
  - Creare competenze
- Svantaggi
  - Spese per tools, sviluppo metodologie
  - Molto difficile arrivare ad una accuratezza elevata, serve molto tempo per formare il personale

Verifica della sicurezza: terza parte

- Vantaggi:
  - Utilizzo di personale dedicato a queste attività con competenze tecniche allo stato dell'arte
  - Risultati più approfonditi
- Svantaggi
  - Poco scalabile su centinaia di applicazioni in poco tempo

Code Review vs Application Testing

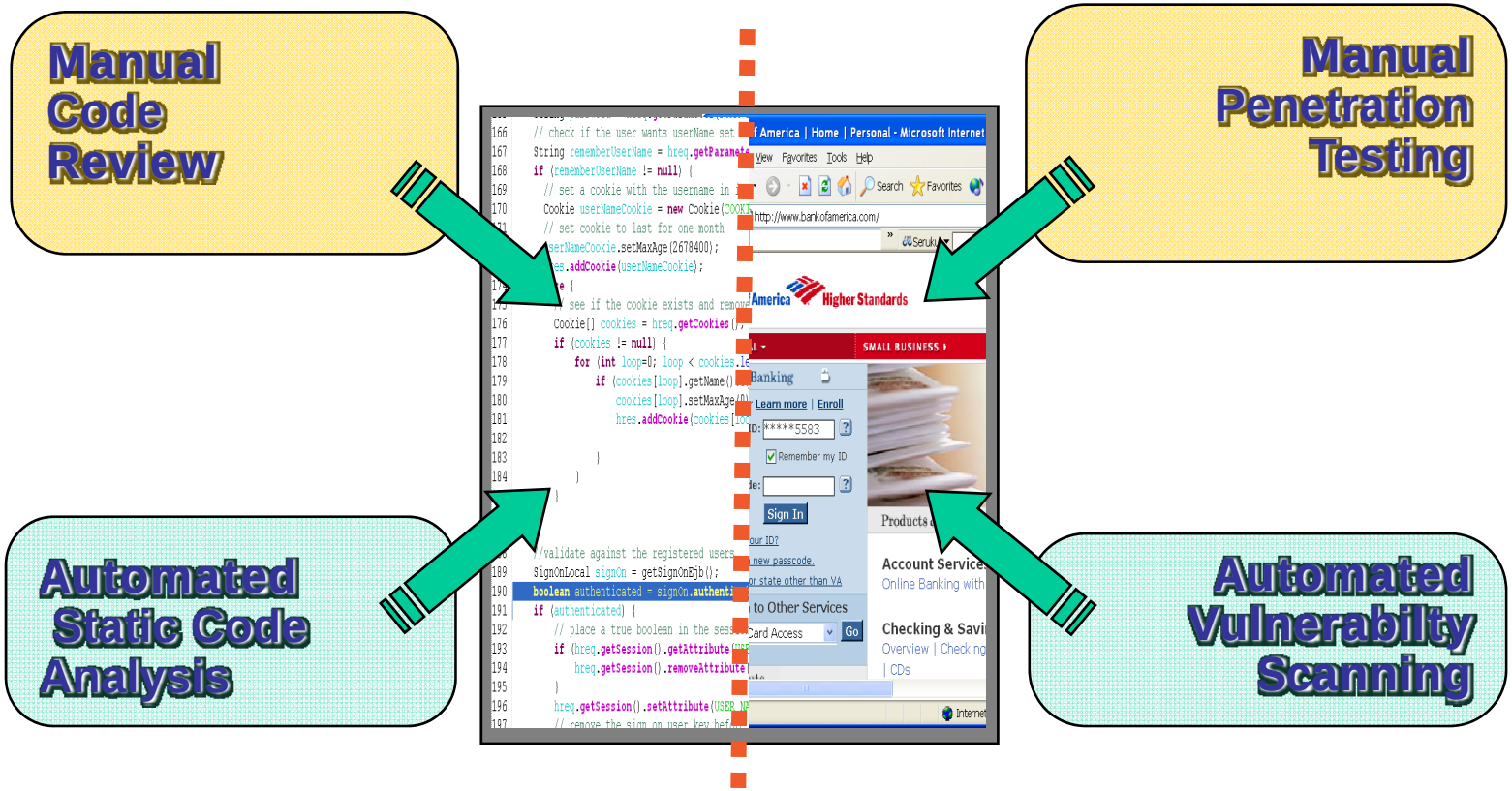
- **Secure Code Review:** l'attività di secure Code Review consiste nell'analisi di sicurezza del codice sorgente dell'applicativo linea per linea: viene anche chiamato test di tipo white box, per sottolineare il fatto che chi esegue la verifica ha a disposizione la conoscenza completa dell'applicativo (insieme dei sorgenti).
- **Web Application Penetration Testing (WAPT):** l'attività di Web Application Penetration Testing consiste nell'effettuare una simulazione reale di un attacco informatico all'applicativo in oggetto al fine di valutarne l'effettivo livello di sicurezza. Tale test, viene chiamato di tipo black box in quanto in questa circostanza chi compie l'analisi non ha a disposizione nessuna conoscenza sul software, e vuole garantire che non siano presenti problematiche di sicurezza prima del deploy in esercizio.

Manuale vs Automatico

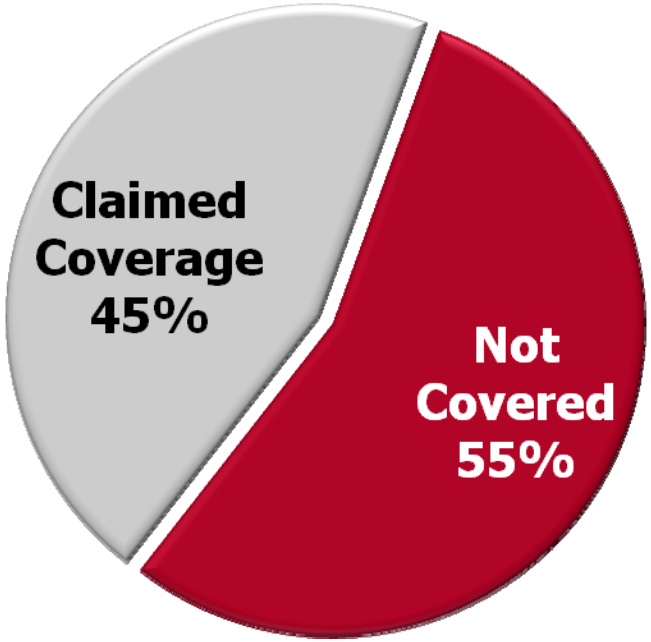
Trovare vulnerabilità nel Codice Sorgente (White Box Testing)

La combinazione delle 4 tecniche produce I risultati migliori

Trovare vulnerabilità nelle applicazioni sviluppate (Black Box Testing)



Tools – At Best 45%



- MITRE found that all application security tool vendors' claims put together cover only 45% of the known vulnerability types (over 600 in CWE)
- They found very little overlap between tools, so to get 45% you need them all (assuming their claims are true)

SECURITYINNOVATION · FORTIFY SOFTWARE · PALAMIDA · watchfire · Klocwork · Secure Software · AppSIC · CENZIC · CORE SECURITY TECHNOLOGIES · PARASOFT · Coverity · proServices · SofCheck · SPI DYNAMICS · VERACODE · OUNCE LABS

Conclusione

- Come affrontare il tema della **Web application security** nelle PA:
  - Progettare applicativi seguendo una **standard riconosciuti** in modo che il servizio non sia vulnerabile a possibili attacchi web.
  - Concepire la **sicurezza by-design** e non come semplice add-on
  - Fattore chiave nello sviluppo in **qualità** di applicazioni
  - Implementare un **programma definito di Software Assurance** con linee guida standard, percorsi di formazione, processi di security integrati del ciclo di vita di sviluppo del software

Grazie!

Domande?

matteo.meucci@owasp.org

matteo.meucci@mindedsecurity.com