OWASP – Beyond the Top 10

André Rochefort
TELUS Security Assessment Services
Sr. Consultant
andre.rochefort@telus.com

“All programmers are playwrights and all computers are lousy actors.” (unknown)
What is this about?

OWASP Present & Future Solutions:

• Flagship Projects
• Labs Projects
• Incubator Projects
Communities

A Vision for OWASP

Outreach
Projects
Stakeholders
Focus
Support
Platform

 Builders
 Breakers
 Defenders

Global Committees
Board

OWASP
Target Audiences

1) Students* and AppSec neophytes
   • The ones eager to learn

2) Developers and IT Security Administrators
   • The ones anxious to defend

3) AppSec Professionals & Community
   • The ones making all the noise
Why?

• Raise awareness
• Call to Arms / Engage
• Sharpen those soft skills

“No man is exempt from saying silly things; the mischief is to say them deliberately.” - Michel de Montaigne
# WebAppSec Resources vs. Backlog

<table>
<thead>
<tr>
<th>Assumptions</th>
<th>Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of websites on the public Internet</td>
<td>672,985,183</td>
</tr>
<tr>
<td>Number of hours a webappsec pen-tester takes to assess the average website</td>
<td>16</td>
</tr>
<tr>
<td>Number of work hours in the average year</td>
<td>2,000</td>
</tr>
<tr>
<td>Number of people working as webappsec pen-testers today</td>
<td>300,000</td>
</tr>
<tr>
<td>Number of scans required per year</td>
<td>4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Results</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of web application testers needed to test the entire Internet manually:</td>
<td>21,535,526</td>
</tr>
<tr>
<td>Number we will have to hire to get full global coverage:</td>
<td>21,235,526</td>
</tr>
<tr>
<td>Number of sites that won’t get assessed by hand due to shortage:</td>
<td>663,610,183</td>
</tr>
<tr>
<td>Percent of all Sites Covered by Manual Assessments:</td>
<td>1.4%</td>
</tr>
</tbody>
</table>
The Top 10
(and other news)
Recent OWASP News

• The 2013 WebAppSec Top 10 Launched

• Source Code Analyzer Coverity joins OWASP

• State of the Community
<table>
<thead>
<tr>
<th>OWASP Top 10 – 2010 (Previous)</th>
<th>OWASP Top 10 – 2013 (New)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1 – Injection</td>
<td>A1 – Injection</td>
</tr>
<tr>
<td>A3 – Broken Authentication and Session Management</td>
<td>A2 – Broken Authentication and Session Management</td>
</tr>
<tr>
<td>A2 – Cross-Site Scripting (XSS)</td>
<td>A3 – Cross-Site Scripting (XSS)</td>
</tr>
<tr>
<td>A4 – Insecure Direct Object References</td>
<td>A4 – Insecure Direct Object References</td>
</tr>
<tr>
<td>A6 – Security Misconfiguration</td>
<td>A5 – Security Misconfiguration</td>
</tr>
<tr>
<td>A7 – Insecure Cryptographic Storage – Merged with A9</td>
<td>A6 – Sensitive Data Exposure</td>
</tr>
<tr>
<td>A8 – Failure to Restrict URL Access – Broadened into</td>
<td>A7 – Missing Function Level Access Control</td>
</tr>
<tr>
<td>A5 – Cross-Site Request Forgery (CSRF)</td>
<td>A8 – Cross-Site Request Forgery (CSRF)</td>
</tr>
<tr>
<td>&lt;buried in A6: Security Misconfiguration&gt;</td>
<td></td>
</tr>
<tr>
<td>A10 – Unvalidated Redirects and Forwards</td>
<td>A9 – Using Known Vulnerable Components</td>
</tr>
<tr>
<td>A9 – Insufficient Transport Layer Protection</td>
<td>Merged with 2010-A7 into new 2013-A6</td>
</tr>
</tbody>
</table>
### Additional Risks to Consider

The Top 10 cover a lot of ground, but there are many other risks you should consider and evaluate in your organization. Some of these have appeared in previous versions of the Top 10, and others have not, including new attack techniques that are being identified all the time. Other important application security risks (in alphabetical order) that you should also consider include:

- Clickjacking
- Concurrency Flaws
- **Denial of Service** (Was 2004 Top 10 – Entry 2004-A2)
- Expression Language Injection (CWE-917)
- Information Leakage and Improper Error Handling (Was part of 2007 Top 10 – Entry 2007-A6)
- Insufficient Anti-automation (CWE-799)
- Insufficient Logging and Accountability (Related to 2007 Top 10 – Entry 2007-A6)
- Lack of Intrusion Detection and Response
- Malicious File Execution (Was 2007 Top 10 – Entry 2007-A3)
- Mass Assignment (CWE-915)
- User Privacy

---

- e.g. **WS Amplification**
- e.g. **Facebook Shadow Profiles**
- e.g. **PRISM**
Other Top 10s

- Top 10 Mobile Risks (refresh: 2013)
- Top 10 Mobile Security Controls
- Top 10 Defenses
- Top 10 Cloud Risks

Also:

- Alternative classification schemes, e.g. The Seven Pernicious Kingdoms
Gartner Magic Quadrant
State of the Community

- Mark Curphey on OWASP; Seconauts, and Security Tools for Developers
- OWASP Top 10 – 9 Too Many?
- Dini Cruz and OWASP in 2014
- Pushing for more activity in T.O.
Top 5 Developer Fears

(from: Itworld/StackOverflow)

1. Screwing up*
2. Losing their jobs
3. No longer liking the job
4. Learning new technologies
5. Incompetent Management/Coworkers
Beyond the Top 10
The Inventory*

- Resources for WebAppSec Training
- Secure Coding Materials, APIs, SCAs
- Tools for Vulnerability Mitigation, Discovery
- Miscellany in between
The Learning Curve

OWASP Tools for WAS Education:

• Tutorials / Exercised-based Training
• Vulnerable Web Applications
• Books!

Download the PDFs free or buy hardcopies and support OWASP
OWASP WebGoat

Solution Videos

Stage 4
Stage 4: Block Stored XSS using Output Encoding.

THIS LESSON ONLY WORKS WITH THE DEVELOPER VERSION OF WEBGOAT

Implement a fix to block XSS after it is read from the database. Repeat stage 3. Verify that 'David' is not affected by Bruce's profile attack.
OWASP WebGoat

- Platform variants: Java, .NET and Rails (coming), Desktop(+Top 5?)
- Mobile variants: iGoat/GoatDroid
- Content-rich; Roll-Your-Own Lessons
- Video tutorials online & downloadable
- Report Cards, Challenge Mode
OWASP Mutillidae II: Web Pwn in Mass Production

Version: 2.5.11  Security Level: 0 (Hosed)  Hints: Enabled (1 - 5cr1pt K1dd1e)  Not Logged In

OWASP Top 10
A1 - SQL Injection
A2 - Cross Site Scripting (XSS)
A3 - Broken Authentication and Session Management
A4 - Insecure Direct Object References
A5 - Cross Site Request Forgery (CSRF)
A6 - Security Misconfiguration
A7 - Insecure Cryptographic Storage
A8 - Failure to Restrict URL Access
A9 - Insufficient Transport Layer Protection
A10 - Unvalidated Redirects and Forwards

Web Services
A1 - Other Injection

HTML 5
A2 - HTML Injection (HTMi)
A3 - HTMLi via HTTP Headers
A4 - HTMLi via DOM Injection
A5 - HTMLi via Cookie Injection
A6 - Frame Source Injection
A7 - Command Injection
A8 - JavaScript Injection
A9 - HTTP Parameter Pollution
A10 - Cascading Style Injection
A11 - JavaScript Object Notation (JSON) Injection
A12 - Buffer Overflow
A13 - Parameter Addition
A14 - XML External Entity Injection

Others
A1 - Deliberately Vulnerable Web Pen-Testing Application

Support Email Address
Announcements
Feature Requests

Site hacked...err...quality tested with Firefox, Burp-Suite and these Mozilla Add-ons

PHP MyAdmin Console
OWASP Mutillidae 2

- Includes HTML5-oriented lessons
- Plenty of content (lessons, tutorials)
- Video guides available (YouTube)
- Gamified! Keeps track of your score
- PHP, requires (L|W|M)AMP stack
Welcome to Bricks!

Bricks is a web application security learning platform built on PHP and MySQL. The project focuses on variations of commonly application security issues. Each 'Brick' has some sort of security issue which can be leveraged manually or using automated. The mission is to 'Break the Bricks' and thus learn the various aspects of web application security.

Bricks is a completely free and open source project brought to you by OWASP. The complete documentation and instruction can be accessed or downloaded for free. Bricks are classified into three different sections: login pages, file upload pages and cont...

File Upload pages

Each file upload page has its own security mechanisms. Some pages break them, upload shell scripts, execute them and gain access.

Upload #1
Simple file upload.

Upload #2
Challenge #6
What are Insecure Direct Object References?

Imagine a web page that allows you to view your personal information. The web page that shows the user their information is generated based on a user ID. If this page was vulnerable to **insecure Direct Object References**, an attack would be able to modify the user identifier to reference any user object in the system. Insecure Direct Object References occur when an application references an object by its actual ID or name. This object that is referenced directly is used to generate a web page. If the application does not verify that the user is allowed to reference this object, then the object is *insecurely referenced*.

Attacks can use insecure object references to compromise any information that can be referenced by the parameter in question. In the above example, the attacker can access any user's personal information.

The severity of insecure direct object references varies depending on the data that is been compromised. If the...
Guided Lessons

e-Learning Project (CBT)

OWASP Hackademic

(live version: http://hackademic1.teilar.gr/)

http://vicnum.ciphertechs.com/ (Games!)

http://google-gruyere.appspotspot.com/

http://www.hackertest.net/

Advanced: https://www.hacking-lab.com/about/ (english language issues)
*Vulnerable Web Apps

(*intentionally!*)

• OWASP Broken Web Apps (VM)
• Damn Vulnerable Web Application
• KILL ALL THE VENDOR’S SITES! (live)
• OWASP SiteGenerator (RIP)
• Build your own, then break it!
TRY TO HACK THEM ALL!
More from OWASP

• Book: WebGoat and WebScarab
• The AppSec Tutorial Series (Videos):
  • [https://www.owasp.org/index.php/OWASP_Appsec_Tutorial_Series](https://www.owasp.org/index.php/OWASP_Appsec_Tutorial_Series)
• Cheat Sheets
• Book: Securing WebGoat with ModSecurity
Tangent: Visualization

- Tilt (DEMO!)
- Logstalgia (DEMO!)
- glTail (video!)
- Kinectaploit (video!)
- http://secviz.org/
psDoom
Defending the Code

• Guides, Guides and More Guides
• Enterprise Security API
• AntiSAMY
• Source Code Analyzers
• Java Dependency Checker
Guides! (ick, PDFs)

- Web Application Secure Development Guide
- Code Review Guide (2.0 underway)
- Testing Guide
- Software Assurance Maturity Model (SAMM)
- Periodic Table of Vulnerabilities
- Application Security Verification Standard
Security Requirements

Network security includes the system of computers, routers, cables, switches and wireless access points. It is the entire system of transport and storage technologies.

7.1 Are networks segregated physically and/or logically to separate systems containing personal information from public networks such as the Internet? [YES / NO]

7.2 Where a local area network containing personal information is connected to a public network, does the organization use perimeter defence safeguards (e.g. firewalls, routers, intrusion detection or prevention systems, anti-virus/anti-spyware software, etc.) to mediate all traffic and to protect systems that are accessible from the Internet? [YES / NO]

7.3 Are systems and their software “hardened” (e.g. applications)

7.4 Are ports closed/services are not exposed?

7.5 Are these safe?

---

V2 - Authentication Verification Requirements

The Authentication Verification Requirements define a set of requirements for generating and handling account credentials safely. The table below defines the corresponding verification requirements that apply for each of the four verification levels.

Table 2 - OWASP ASVS Authentication Requirements (V2)

<table>
<thead>
<tr>
<th>Verification Requirement</th>
<th>Level 1A</th>
<th>Level 1B</th>
<th>Level 2A</th>
<th>Level 2B</th>
<th>Level 3</th>
<th>Level 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>V2.1 Verify that all pages and resources require authentication except those specifically intended to be public.</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>V2.2 Verify that all password fields do not echo the user’s password when it is entered, and that password fields (or the forms that contain them) have autocomplete disabled.</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>V2.3 Verify that if a maximum number of authentication attempts is exceeded, the account is locked for a period of time long enough to deter brute force attacks.</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
<tr>
<td>V2.4 Verify that all authentication controls are enforced on the server side.</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
<td>✅</td>
</tr>
</tbody>
</table>
OWASP ASVS

Flagship Project

*At higher levels in ASVS, the use of tools is encouraged. But to be effective, the tools must be heavily tailored and configured to the application and framework in use.*

A standard to verify a web app’s security

Application- and lifecycle- independent

OWASP ASVS Levels

1  2  3  4
OWASP Cornucopia
Spoofing
An attacker could go after the way credentials are updated or recovered (account recovery doesn’t require disclosing the old password).

Tampering
An attacker can change parameters over a trust boundary and after validation (for example, important parameters in a hidden field in HTML, or passing a pointer to critical memory).

Information Disclosure
An attacker can read the entire channel because the channel (say, HTTP or SMTP) isn’t encrypted.

Don’t tell anyone, but...
d0x3d!

A network security game

[about]

d0x3d! is a board game designed to introduce a diverse body of students to network security terminology, attack & defend mechanics, and basic computer

[n3ws]

Check out our article with Control-Alt-Hack’s Tamara Denning on tabletop-gaming in security, appearing in the May-June issue of IEEE Security & Privacy.
Game includes:

- 1 rulebook
- 3 dice
- 156 game cards
  - 16 Hacker cards
  - 56 Mission cards
  - 72 Entropy cards
  - 12 Attendance cards
- 58 Hacker Cred tokens
- 42 Money tokens
Daily Crossword

Web Application Security

OWASP via MyAppSecurity
Not hard enough?

RegEx
Crossword
FTOMGWTF
OWASP ESAPI

• FREE Security Control Library
• Reference implementations included
• Extensible, customizable, mature*
• Support includes Java, .NET, PHP, ...
• AppSensor integration

“Good artists copy; great artists steal”
OWASP AntiSAMY

- Policy-based HTML/CSS input validator
- Support includes Java and .NET
- Sample policies available
- PHP: use HTMLPurifier instead
- Sadly, dormant.
OWASP YASCA
• Yet Another Source/Static Code Analyzer
• Frontend to Lint, FindBugs, ClamAV,...
• Pattern-matching engine
• Still in active development
OWASP Code Crawler

```java
private String computeHash(String password, String passwordSalt) throws Exception {
    byte[] buffer = null;
    byte[] key = null;
    byte[] hash = null;
    MessageDigest md5 = null;
    String hashEncoded = null;
    String salt = (passwordSalt.split("\$"))[0];
    try {
        buffer = URLEncoder.encode(password, "ISO-8859-1").getBytes("ISO-8859-1");
        key = URLEncoder.encode(salt, "ISO-8859-1").getBytes("ISO-8859-1");
        md5 = MessageDigest.getInstance("MD5");
        md5.update(buffer);
        hash = md5.digest(key);
        hashEncoded = URLDecoder.decode(new String(hash, "ISO-8859-1"), "ISO-8859-1");
    }
    catch (UnsupportedEncodingException uae) {
        // Should never happen, as the encoding is a string literal
        throw new Exception("Error computing hash.");
    }
}
```

<table>
<thead>
<tr>
<th>Threat Analysis</th>
<th>Repudiation</th>
<th>Keyword</th>
<th>MD5</th>
</tr>
</thead>
<tbody>
<tr>
<td>STRIDE Category</td>
<td>Repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>STRIDE Control</td>
<td>Non-Repudiation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Threat Description</td>
<td>If cryptography is used then is a strong enough cipher used i.e. AES or 3DES. What size key is used, the larger the better. Where is hashing performed. Are password that are being persisted hashed, they should be. How are random numbers generated? Is the PRNG &quot;random enough&quot;?</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
OWASP CodeCrawler

• Static SCA supporting Java and .NET
• Last Update: April 2010 (3-person team)
• RegEx filtering; basic, configurable pattern matching
• No Data Flow validation; Windows Only
• High False Positive/False Negative potential
Defending the Web App

ModSecurity Core Rule Set (new release July 2)

AppSensor (App-based IDS)

More WAF projects on the horizon
Hunting for Vulnerability

- WebScarab
- Zed Attack Proxy
- JoomScan and CMS Scan
- WebSlayer with Skanda
- O2 platform
OWASP WebScarab

Classic!
OWASP Zed Attack Proxy
OWASP Mantra

Tools

Information Gathering

- **Flagfox** Displays a flag icon indicating the current webserver's physical location with many additional features.
- **JSView** Get straight access to scripts and stylesheets included in the current web page.
- **PassiveRecon** Perform passive discovery of target resources utilizing publicly available information.
- **Wappalyzer** Uncovers underlying technologies used on websites like CMS, e-commerce systems, JavaScript frameworks, analytics tools etc..
- **View Dependencies** Shows you all the files which were loaded to show the current page.
- **Link Sidebar** View, search and test hyperlinks in a web page.

Application Auditing

- **Hackbar** Simple security audit / Penetration test tool.

Editors

- **JSView** Get straight access to scripts and stylesheets included in the current web page. View the source code external stylesheets and javascripts.
- **Firebug** Edit, debug, and monitor CSS, HTML, and JavaScript live in any web page.

Proxy

- **HTTP Fox** A built in local proxy for analyzing traffic.
- **FoxyProxy** A proxy management tool with ability to switch between multiple proxies with few clicks.
- **Proxy Tool** A proxy management tool with lots of additional features to enhance the privacy.

Network Utilities

- **FireFTP** FTP/SFTP Client which provides intuitive access to FTP/SFTP servers.
Includes FireCat
Many toolbars behave like malware
Remove unwanted toolbars with avast! Browser Cleanup
OWASP OWTF

Offensive Web Testing Framework

GASP! VIDEO BREAK!

https://www.owasp.org/index.php/OWASP_OWTF
Pantera Web Assessment Studio (WAS) Project: OWASP ASSESSMENT

Last 7 Requests

HTTP/1.1 200 OK
[2006-08-18 | 17:46:07] 🤘

HTTP/1.1 200 OK
[2006-08-18 | 17:46:07] 🤘

HTTP/1.1 200 OK
[2006-08-18 | 17:46:07] 🤘

O2 Platform
O2 Platform

**Warning:**

Tangent

**Problem:**
BlackBox: Easily create XSS PoCs that are specific to the application and are much more than the ALERT pop-up box that nobody outside the WebAppSecurity space understand’s it implication.

**Solution:**
O2 :)

**No More**

**With Security Findings**
WARNING: TANGENT

• Jon McCoy @ SecTor 2012
• <video excerpt>
Swiss Army Knives

- OWASP Mantra OS (Mobile: MobiSec)
- Samurai Web Testing Framework

Alternatives:
- Kali (aka BackTrack)
- Fedora Security Spin
Incubators and More

• iSABEL Proxy Server, NAXSI, WAF Project
• Xenotix XSS Framework vs. XSSer, X5s
• Security Tools for Developers
• Java HTML Sanitizer (released)
• S.T.I.N.G. For Security Requirements?
• VaultDB vs Scytale (DBMS crypto-proxy)
Project Gaps?

• Lots of duplication; re-inventing the wheel
• Inconsistent Quality, no unity in delivery
• No visualization projects; forensics a stub
• Fragmentation; resources spread thin
• Over-promising; under-delivering
• Solutions?
# Google Summer of Code 2013

List of projects accepted into Google Summer of Code 2013

<table>
<thead>
<tr>
<th>Student</th>
<th>Title</th>
<th>Organization</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abdelhadi</td>
<td>ZAP Proxy : CMS Scanner</td>
<td>OWASP</td>
</tr>
<tr>
<td>Alessandro Fanio González</td>
<td>OWASP OWTF - Unit Test Framework</td>
<td>OWASP</td>
</tr>
<tr>
<td>Ankush Jindal</td>
<td>OWASP OWTF - Multiprocessing</td>
<td>OWASP</td>
</tr>
<tr>
<td>Assem Chelli</td>
<td>OWASP OWTF - Reporting</td>
<td>OWASP</td>
</tr>
<tr>
<td>Bharadwaj Machiraju</td>
<td>OWASP OWTF - INBOUND PROXY WITH MIT</td>
<td>OWASP</td>
</tr>
<tr>
<td>Cosmin Stefan</td>
<td>Enhanced HTTP Session Handling and users</td>
<td>OWASP</td>
</tr>
<tr>
<td>Daniel Kvist</td>
<td>Plugin api and plugin actions interface in OWASP</td>
<td>OWASP</td>
</tr>
<tr>
<td>Mihai Pitu</td>
<td>OWASP ModSecurity CRS - Port to Java</td>
<td>OWASP</td>
</tr>
<tr>
<td>Pulasthi Mahawithana</td>
<td>OWASP ZAP - SAML 2.0 Support</td>
<td>OWASP</td>
</tr>
<tr>
<td>Rahul Chaudhary</td>
<td>OWASP PHP Security Project</td>
<td>OWASP</td>
</tr>
<tr>
<td>Rauf Butt</td>
<td>ZAP - Exploring Advanced reporting using</td>
<td>OWASP</td>
</tr>
</tbody>
</table>
Go, Toronto, Go!

- Chapter participation appears to be on the rise
- Tremendous amount of infosec talent in the GTA and surrounding areas
- IRC? Reddit? Hackernews?
- Anyone need an opening act next time?
Q & A

Bookmark: http://owasp.blogspot.ca/
Thank you
THE PERFECT MARTINI

1. Pour gin, vermouth, and olives into the trash where they belong.
2. Drink whiskey

I always keep a supply of stimulant handy in case I see a snake--which I also keep handy.

W. C. Fields (1880 - 1946)