OWASP Top 10 Vulnerabilities: Panel Discussion

Sebastien Deleersnyder
CISSP
Sep, 2005
sdl@ascure.com

OWASP

Agenda

- Panel Introduction
- OWASP Top 10
- Panel Discussion

Agenda

- Panel Introduction
- OWASP Top 10
- Panel Discussion

Panel Introduction

- Erwin Geirnaert, Security Innovation
- Dirk Dussart, Belgian Post
- Eric Devolder, Mastercard
- Herman Stevens, Ubizen
- Frank Piessens, KU Leuven

Agenda

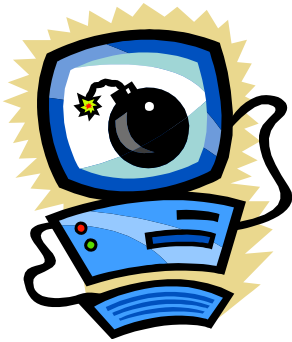
- Introduction
- OWASP Top 10
- Panel Discussion

OWASP Top 10

A1	**Unvalidated Input**
A2	**Broken Access Control**
A3	**Broken Authentication and Session Management**
A4	**Cross Site Scripting (XSS) Flaws**
A5	**Buffer Overflows**
A6	**Injection Flaws**
A7	**Improper Error Handling**
A8	**Insecure Storage**
A9	**Denial of Service**
A10	**Insecure Configuration Management**

1. Unvalidated Input

- HTTP requests from browsers to web apps
  - URL, Querystring, Form Fields, Hidden Fields, Cookies, Headers
  - Web apps use this information to generate web pages
- Attackers can modify anything in request
  - WebScarab
- Key Points:
  - Check before you use anything in HTTP request
  - Canonicalize before you check
  - Client-side validation is irrelevant
  - Reject anything not specifically allowed
    - Type, min/max length, character set, regex, min/max value...



2. Broken Access Control

- Access control is how you keep one user away from other users' information
- The problem is that many environments provide authentication, but don't handle access control well
  - Many sites have a complex access control policy
  - Insidiously difficult to implement correctly
- Key Points
  - Write down your access control policy
  - Don't use any "id's" that an attacker can manipulate
  - Implement access control in a centralized module

3. Broken Authentication and Session Management

- Authentication
  - Handling credentials across client-server gap
  - Backend authentication credentials too
- Session Management
  - HTTP is a "stateless" protocol. Web apps need to keep track of which request came from which user
  - "Brand" sessions with an id using cookie, hidden field, URL tag, etc…
- Key Points
  - Keep credentials secret at all times
  - Use only the random sessionid provided by your environment

4. Cross-Site Scripting (XSS) Flaws

- Web browsers execute code sent from websites
  - Javascript
  - Flash and many others haven't really been explored
- But what if an attacker could get a website to forward an attack!
  - Stored – web application stores content from user, then sends it to other users
  - Reflected – web application doesn't store attack, just sends it back to whoever sent the request
- Key Points
  - Don't try to strip out active content – too many variations. Use a "positive" specification.

5. Buffer Overflows

- Web applications read all types of input from users
  - Libraries, DLL's, Server code, Custom code, Exec
- C and C++ code is vulnerable to buffer overflows
  - Input overflows end of buffer and overwrites the stack
  - Can be used to execute arbitrary code
- Key Points
  - Don't use C or C++
  - Be careful about reading into buffers
  - Use safe string libraries correctly

6. Injection Flaws

- Web applications involve many interpreters
  - OS calls, SQL databases
- Malicious code
  - Sent in HTTP request
  - Extracted by web application
  - Passed to interpreter, executed on behalf of web app
- Key Points
  - Use extreme care when invoking an interpreter
  - Use limited interfaces where possible (PreparedStatement)
  - Check return values

7. Improper Error Handling

- Errors occur in web applications all the time
  - Out of memory, too many users, timeout, db failure
  - Authentication failure, access control failure, bad input
- How do you respond?
  - Need to tell user what happened (no hacking clues)
  - Need to log an appropriate (different) message
  - Logout, email, pager, clear credit card, etc…
- Key Points:
  - Make sure error screens don't print stack traces
  - Design your error handling scheme
  - Configure your server

8. Insecure Storage

- Use cryptography to store sensitive information
  - Algorithms are simple to use, integrating them is hard
- Key Points
  - Do not even think about inventing a new algorithm
  - Be extremely careful storing keys, certs, and passwords
  - Rethink whether you need to store the information
  - Don't store user passwords – use a hash like SHA-256
- The "master secret" can be split into two locations and assembled
  - Configuration files, external servers, within the code

9. Denial of Service

- Difference between attack and ordinary traffic?
  - IP address filtering?
  - DDOS, Slashdotted?
  - Account lock-outs
  - Abuse error, consume resources
- Key Points
  - Limit resource allocation
  - Use quotas
  - Avoid “expensive” operations
  - Graceful handling of errors

10. Insecure Configuration Management

- All web and application servers have many security-relevant configuration options
  - Default accounts and passwords
  - Unnecessary default, backup, sample apps, libraries
  - Overly informative error messages
  - Misconfigured SSL, default certificates, self-signed certs
  - Unused administrative services
- Key Points:
  - Keep up with patches (Code Red, Slammer)
  - Use Scanning Tools (Nikto, Nessus)
  - Harden your servers!

OWASP Top 10 Success

- E.g. referenced by payment card industry, Federal Trade Commission, US gov.
- Used by Sprint, IBM Global Services, Cognizant, Foundstone, Strategic Security, Bureau of Alcohol, Tobacco, and Firearms (ATF), Sun Microsystems, British Telecom, Swiss Federal Institute of Technology, Sempra Energy, Corillian Corporation, A.G. Edwards, Texas Dept of Human Services, Predictive Systems, Price Waterhouse Coopers, Best Software, Online Business Systems, ZipForm, Contra Costa County, SSP Solutions, Recreational Equipment, Inc. (REI), Zapatec, Cboss Internet, Samsung SDS (Korea), Norfolk Southern, Bank of Newport
- part of curriculum at Michigan State University and the University of California at San Diego

Agenda

- Introduction
- OWASP Top 10
- Panel Discussion

Panel Discussion

- Does the Top 10 improve web application security?
- Are we talking vulnerabilities, solutions or threats?
- Can we base our best practices / standards on the Top 10?
- Should the Top 10 be incorporated in curricula?
- How to test your web site security on the Top 10?
- Do we need localized versions of the Top 10?
- Is the OWASP Top 10 still necessary?
- How to update the Top 10?
  - Other format?
  - Survey?
  - …