Secure Coding, some simple steps help.
About Me

- Steven van der Baan
- Dutch
- 7Safe, part of PA Consulting Group
  - Developer
  - Pentester
  - Consultant
- CISSP, OSCP
“It's amazing how many drivers think the brakes are for slowing the car down”

“Brakes allow you to travel faster because you have the power to stop.”
Who.. ..with what role..
..may do..
..what rights.

in which process?
## Specific vulnerabilities for each principle

<table>
<thead>
<tr>
<th>Principles</th>
<th>OWASP top 10</th>
<th>WhiteHatSec top 10</th>
<th>Sans top 25</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Input Validation</strong></td>
<td>Cross Site Scripting, Injection Flaws, Malicious File Execution</td>
<td>Cross Site Scripting, SQL Injection, Content Spoofing</td>
<td>Improper Input Validation, Failure to Preserve SQL Query Structure, Failure to Preserve Web Page Structure, Failure to Preserve OS Command Structure, Failure to Constrain Operations within the Bounds of a Memory Buffer, Failure to Control Generation of Code, Client-Side Enforcement of Server-Side Security</td>
</tr>
<tr>
<td><strong>Output Encoding</strong></td>
<td>Cross Site Scripting</td>
<td>Cross Site Scripting</td>
<td>Improper Encoding or Escaping of Output, Failure to Preserve Web Page Structure</td>
</tr>
<tr>
<td><strong>Error Handling</strong></td>
<td>Information Leakage and Improper Error Handling</td>
<td>Information Leakage</td>
<td>Error Message Information Leak</td>
</tr>
<tr>
<td><strong>Session Management</strong></td>
<td>Broken Authentication and Session management, Cross Site Request forgery</td>
<td>Cross Site Request Forgery</td>
<td>Cross Site Request Forgery, Use of Insufficient Random Values</td>
</tr>
<tr>
<td><strong>Secure Communications</strong></td>
<td>Insecure Communications</td>
<td></td>
<td>Use of a Broken or Risky Cryptographic Algorithm, Cleartext Transmission of Sensitive Information, Use of Insufficiently Random Values</td>
</tr>
<tr>
<td><strong>Secure Resource Access</strong></td>
<td>Insecure Direct Object Reference, Failure to Restrict URL Access</td>
<td>Predictable Resource Location</td>
<td>External Control of File Name or Path, Untrusted Search Path</td>
</tr>
<tr>
<td><strong>Secure Storage</strong></td>
<td>Insecure Cryptographic Storage</td>
<td></td>
<td>Use of a Broken or Risky Cryptographic Algorithm, Cleartext Transmission of Sensitive Information, External Control of Critical State Data</td>
</tr>
</tbody>
</table>
• Input Validation
• Output Encoding
• Error Handling
• Authentication and Authorisation
• Session Management
• Secure Communications
• Secure Resource Access
• Secure Storage
Input validation
Input validation

• Identify and define the data your application must accept
• Create regEx’s to validate EACH datatype (content and size)
  – For example, a creditcard data type: \d{12,16}$
• Use whitelisting where possible
• Blacklist approach harder and potential less secure
  – Blacklist example, replace single quotes:
    • S.replaceAll(Pattern.quote(""));
    • Matcher.quoteReplacement("");}
Output Encoding
• Identify and define the data your application must output
• Understand where (i.e. in an URL) your data should end up
• Choose the correct output encoding for the data’s destination
  – Proper encoding means this attack:
    • www.example.com/home.html?day=<script>alert(document.cookie)</script>
Becomes
    • day=%3Cscript%3Ealert%28document.cookie%29%3C/script%3E
Error Handling
• Even the best apps will crash at some point, be prepared!
• Crashed/errors can help an attacker if you don’t handle them
• Handle error conditions securely, sanitize the message sent
• No error handling = information leakage

Microsoft OLE DB Provider for ODBC Driver(0x80040E14)
[Microsoft][ODBC SQL Server Driver]
[SQL Server]Invalid column name

/example/login.asp, line 10
Authentication & Authorization
Authentication and Authorisation

OWASP
The Open Web Application Security Project

Membership Card

4297 4392 7156 8735
• Even simple apps often have a need to authenticate users
• Often at least two levels of authorisation
• Need to prevent horizontal and vertical privilege escalation
• Implement strong passwords and management system
• Ensure A+A is secure, not a false sense of security (CAPTCHA?)
• Don’t rely on fields that are easily spoofed (referer field)
Session Management
Session Management

OWASP
The Open Web Application Security Project
• Used to manage authenticated users, no need to re-auth
• You need to make sure your sessionID’s have sufficient entropy
• SessionID’s must not be predictable or reusable
• Never build your own session management, it will fail
• Protect SessionID’s when in transit (i.e. SSL!)
• Issue a new value for sensitive actions (i.e. funds transfer)
Secure Communications
Secure Communications

OWASP
The Open Web Application Security Project

[Image of two orange stick figures using a tin can phone with strings and a photo of two individuals using a vintage cryptographic device]
• Protect data (i.e. CC no, passwords, sessionID’s) in transit
• As with all crypto, don’t create your own
• Don’t use broken protection mechanisms (i.e. SSLv2)
• Don’t just use SSL/TLS for logon pages, protect the session!
• Try to avoid mixing secure and insecure traffic on a page
Secure Resource Access
• Obscurity != security, don’t try to hide sensitive resources

• Understand the users flow through an app, cover weak points
  – T-Mobile didn’t do the above, Paris Hiltons account hacked
Secure Storage
Secure Storage
• Protect data (i.e. CC no, passwords, sessionID’s) when stored
• As with all crypto, DON’T create your own
• Don’t use broken protection mechanisms (i.e. DES)
• Don’t store data in places where you can’t confidently secure it
• Strong protection mechanisms, how strong should it be?