What Causes Software Vulnerabilities?

- flaws in developer's own code
- flaws resulting from interactions with other systems
- production deadlines
- cost limitations
- lack of basic security awareness
- not aware of threats inherent to underlying technologies
- unfamiliar with best mitigation practices

Software Assurance

Responsibilities of the software engineer
- minimize functional flaws
- minimize development time and cost
- maximize performance
- maximize maintainability
- maximize usability
- maximize testability, portability, reusability
- minimize security vulnerabilities
Where are the security breaches?
Note that functional requirements define what the software is suppose to do.

How to be responsible (as a software engineer)

Know your language!
Know your environment!
Follow sound software engineering habitually
Follow sound security engineering habitually
Keep current with relevant attack methods
Test, test, test!
  • test for correctness (what it’s ______________)
  • test for security (what it’s ______________)
What do we know about software engineering?

There are many well defined lifecycle models.

All lifecycle models have activities and processes in common.

In general all of these models largely ignore security.

What has been done to “build security in”? 

SSE-CMM

Microsoft Trustworthy Computing Incremental Approach

SSE-CMM  (ISO/IEC 21827 2008 std)
Systems Security Engineering Capability Maturity Model

Project and Organization Process

- ensure quality
- manage configuration
- manage project risk
- monitor & control technical effort
- plan technical effort
- define systems engineering process
- manage product line evolution
- manage support environment
- provide ongoing knowledge
- coordinate with suppliers

Security Engineering Process

Engineering Process

- specify security needs
- provide security input
- monitor security posture
- administer security control
- coordinate security

Assurance Process

- verify and validate security
- build assurance arguments

Risk Process

- assess threat
- assess vulnerability
- assess impact
- assess risk

buildsecurityin.us-cert.gov/bsi/articles/knowledge/sdlc/326-BSI.html
Microsoft Lifecycle

Design Phase
- design complete

Development Phase
- test plans complete
- code complete
- ship
- security
  - develop security tests
  - security
    - *raise awareness
    - *find & fix vulnerabilities
    - *eliminate bad habits

Test Phase
- ongoing activity

Maintenance Phase

Additional Thoughts (about the process)

Use a separate security team
...and make them responsible.

Secure Analysis
- ask security-related _________
- perform ________________
- establish secure coding _________

Secure Design
- observe secure design ____________

Secure Implementation
- awareness of known attacks
- no ________________

Secure Testing
- reviews for secure design and coding principles
- ________________tests
- ________________tests
- track security bugs
### SDL for Agile

<table>
<thead>
<tr>
<th>Training</th>
<th>Requirements</th>
<th>Design</th>
<th>Implementation</th>
<th>Verification</th>
<th>Release</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>Core Security Training</td>
<td>Establish Security Requirements</td>
<td>Establish Design Requirements</td>
<td>Use Approved Tools</td>
<td>Perform Dynamic Analysis</td>
<td>Create Incident Response Plan</td>
<td>Execute Incident Response Plan</td>
</tr>
<tr>
<td>Create Quality Gates/Bug Bars</td>
<td>Perform Attack Surface Analysis/Reduction</td>
<td>Use Threat Modeling</td>
<td>Deprecate Unsafe Functions</td>
<td>Perform Fuzz Testing</td>
<td>Conduct Final Security Review</td>
<td></td>
</tr>
<tr>
<td>Perform Security and Privacy Risk Assessments</td>
<td>Perform Static Analysis</td>
<td>Create Incident Response Plan</td>
<td></td>
<td>Conduct Attack Surface Review</td>
<td>Certify Release and Archive</td>
<td></td>
</tr>
</tbody>
</table>