Secure Development Lifecycle, The good, the bad and the ugly!
Applications are about information!

• 3 pillars of Information Security:
  – Confidentiality
  – Integrity
  – Availability
User requirements

Business requirements

User requirements

Business rules

Constraints

System requirements

Externe interfaces

Why

What

How

Functional

Non-functional
Development Environment
- WebServer
- Application Server
- Database Server

Test Environment
- WebServer
- Application Server
- Database Server

Production Environment
- WebServer
- Application Server
- Database Server

System Environment
- Internet
- Web
- Application
- Back Office
- Database
- Firewall
- DMZ
- Private Network

Questions:
- System access?
- Database access?
- Error handling?
- User privileges?
- Private Network?
Your security “perimeter” has huge holes at the application layer.

You can’t use network layer protection (firewall, SSL, IDS, hardening) to stop or detect application layer attacks.
An Attacker has 24x7x365 to Attack

Attacker Schedule

The Defender has 20 man days per year to detect and defend
MITRE found that all application security tool vendors’ **claims** put together cover only 45% of the known vulnerability types (695).

They found **very** little overlap between tools, so to get 45% you need them all (assuming their claims are true).
<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>A9 – Using Known Vulnerable Components</td>
</tr>
<tr>
<td>A10 – Unvalidated Redirects and Forwards</td>
<td>A10 – Unvalidated Redirects and Forwards</td>
</tr>
<tr>
<td>A9 – Insufficient Transport Layer Protection</td>
<td>Merged with 2010-A7 into new 2013-A6</td>
</tr>
</tbody>
</table>
OWASP
The Open Web Application Security Project

Diagram:
- **Internet**
- **Business Model**
  - Batch
  - Web Service
  - Presentation Service
  - Application Services
  - Business Services
  - Data Service
  - Integration Service
- **Database**
- **External Systems**
OUR GOAL IS TO WRITE BUG-FREE SOFTWARE. I'LL PAY A TEN-DOLLAR BONUS FOR EVERY BUG YOU FIND AND FIX.

YAHOO! WE'RE RICH!

I HOPE THIS DRIVES THE RIGHT BEHAVIOR.

I'M GONNA WRITE ME A NEW MINIVAN THIS AFTERNOON!
• Comprehensive, Lightweight Application Security Process
  – Centered around 7 AppSec Best Practices
  – Cover the entire software lifecycle (not just development)
• Adaptable to any development process
  – Defines roles across the SDLC
  – 24 role-based process components
  – Start small and dial-in to your needs
Part of the ‘Big 4’

Building Guide
Code Review Guide
Testing Guide

Application Security Desk Reference (ASDR)
• Free and open source
  – Gnu Free Doc License
• Most platforms
  – Examples are J2EE, ASP.NET, and PHP
• Comprehensive
  – Thread Modeling
  – Advise & Best Practices
  – Web Services
  – Key AppSec Area’s:
    • Authorization/Authentication
    • Session Management
    • Data Validation
What it is:

- Examination of developed source code for quality.
- Security = Quality
- Robust & Stable code
- More Expensive
- Can be more Accurate
- Requires unique skill set to do properly

What it isn't:

- Silver Bullet
- Replacement for other security controls
- Replacement for poor application development
- Easy
- Cheap (Not Manual anyways)
Secure Code review process – Operational process

Business Requirements & functional context
Identification of entry and exit points
Transactional analysis
Issue identification & risk rating
Identification of potential solutions
Executive summary & detailed report

Design Risk Analysis
User Risk Analysis
Architecture Risk Analysis
Static Code Analysis
Manual Review
Security Metrics
Industry Leading Practice
Coding Standards
• 1. Frontispiece
• 2. Introduction
• 3. The OWASP Testing Framework
• 4. Web Application Penetration Testing
• 5. Writing Reports: value the real risk
• Appendix A: Testing Tools
• Appendix B: Suggested Reading
• Appendix C: Fuzz Vectors
• Appendix D: Encoded Injection
Part of the ‘Big 4 +1’

**Application Security Desk Reference (ASDR)**
No malicious developers
The design has to be right
The controls have to be right
Scan

Depth – Level of Rigor

Breadth – Number of Requirements

Coverage
Find Vulnerabilities
Using the Running Application

Manual Application Penetration Testing
Automated Application Vulnerability Scanning

Find Vulnerabilities
Using the Source Code

Manual Security Code Review
Automated Static Code Analysis
Part of the ‘Big 4 +2’

OWASP
The Open Web Application Security Project

ASVS
SAMM

Building Guide
Code Review Guide
Testing Guide

Application Security Desk Reference (ASDR)
• Start with the core activities tied to any organization performing software development

• Named generically, but should resonate with any developer or manager
• From each of the Business Functions, 3 Security Practices are defined
• The Security Practices cover all areas relevant to software security assurance
• Supports both lightweight and detailed assessments
• Organizations may fall in between levels (+)
Threat Modeling – The Basics

Threat: Causes harm

Vulnerability: Exploitable weakness

Risk: Chance of harm occurring

Countermeasure: Reduces risk

Asset: Valuable resource
Why start again?

OWASP
The Open Web Application Security Project

Threat

Risk is low

Asset

Countermeasure

Dependency's Threat

Dependency's Countermeasure

Dependency's

Dependency
Applications are about information
  > Confidentiality, Integrity & Availability

Explicit security requirements
  > Make security verifiable!

Security in depth
  > Security considered through the whole application
  > Propagation of credentials

Security by default
  > Who may do what?

More code == more bugs!
Functional Designers & Architects:
  > It is not only about what functionality the application has to supply, it also what it may not!

Engineers:
  > Quality is not just ‘does it work’.

Testers:
  > Security weaknesses are not different from other, functional, bugs. They can be traced down the same way.

Managers:
  > Reserve project time for security
  > Understand security as mandatory value of an application

Security Analyst:
  Involve a security Analyst at the beginning of the design phase.
That's it...

..thank you!