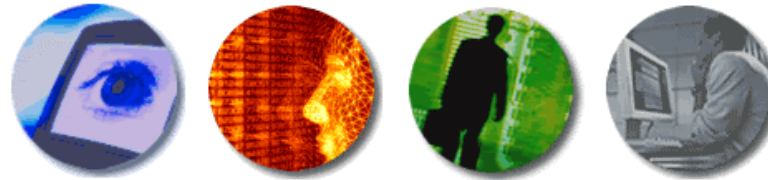




RELIABLE, RESPONSIBLE E-SECURITY



Best Practices for Secure Development

Ron Woerner, CISSP

2003 NebraskaCERT Conference© 2003
Solutionary, Inc.
Solutionary Proprietary and Confidential



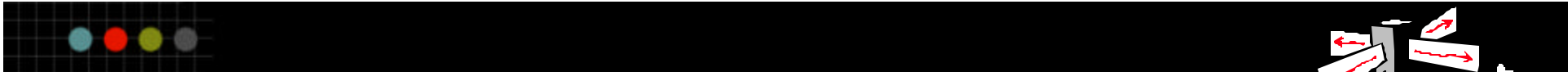
Thoughts

"There are two ways to write error-free programs. Only the third one works."
-Anon.

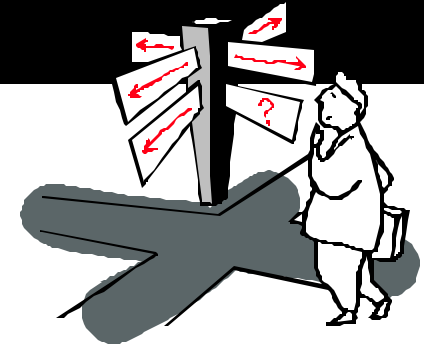
If the developer's would program right in the first place, we wouldn't have all of these security problems.*

So, what can we do to help our developers?

* Not a quote, just what I've heard others say.



Discussion Outline



- General Guidelines for Developers
- Secure Development and Programming
- Security and Software Engineering
- Role-Based Access Control
- Security Links

Please feel free to ask questions, add comments, etc. at any time.



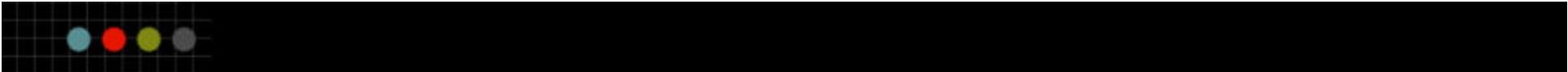
OWASP Top 10 Web Programming Mistakes

1. Unvalidated Parameters
2. Broken Access Control
3. Broken Account & Session Management
4. Cross-Site Scripting (XSS) Flaws
5. Buffer Overflows
6. Command Injection Flaws
7. Error Handling Problems
8. Insecure Use of Cryptography
9. Remote Administration Flaws
10. Web & Application Server Misconfiguration



Attacker's Advantage & Defender's Dilemma[#]

1. The defender must defend all points; the attacker can choose the weakest point.
2. The defender can defend only against known attacks; the attacker can probe for unknown vulnerabilities.
3. The defender must be constantly vigilant; the attacker can strike at will.
4. The defender must play by the rules; the attacker can play dirty.

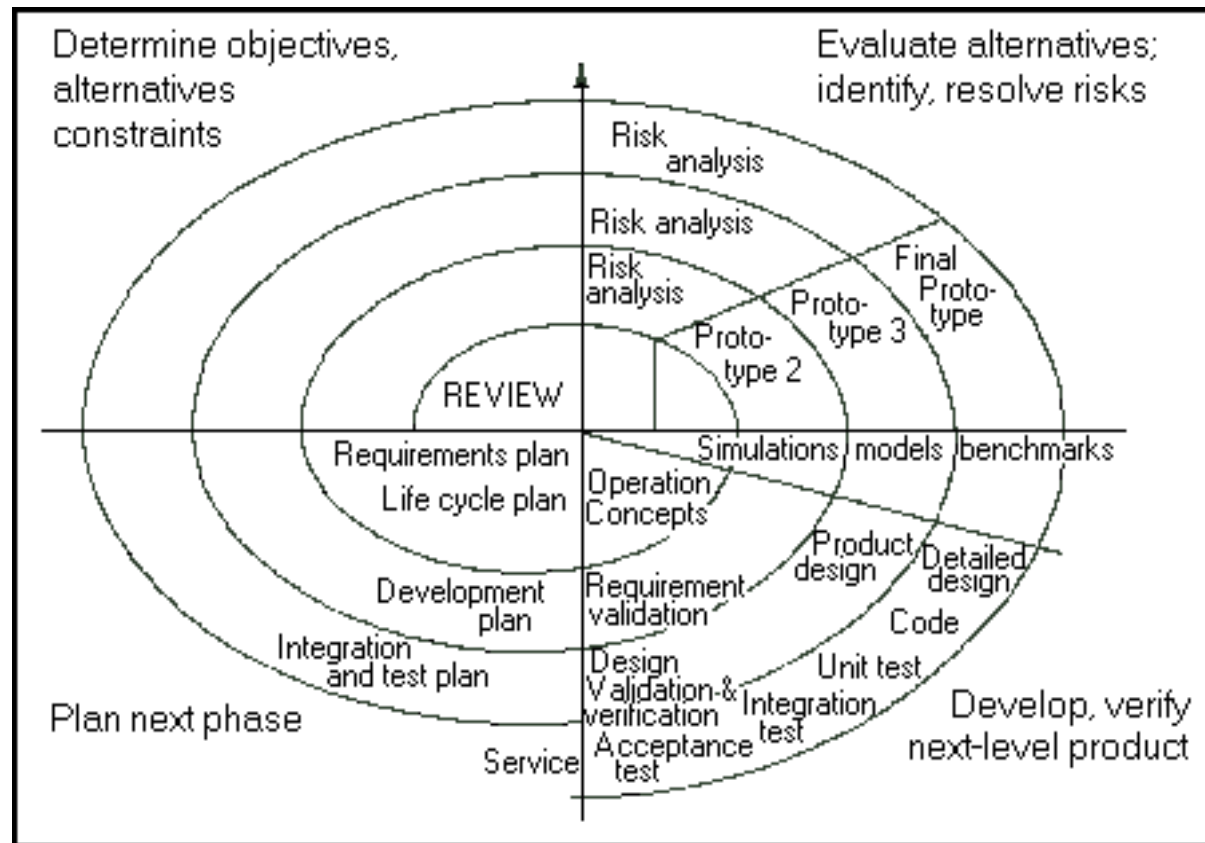


Security and Software Engineering

- All software models have a place for security
 - Analysis & Requirements
 - Design
 - Implementation
 - Testing
 - Operation
- Security must be considered from the beginning
 - DON'T TRY TO ADD IT IN LATER!

Security and Software Engineering

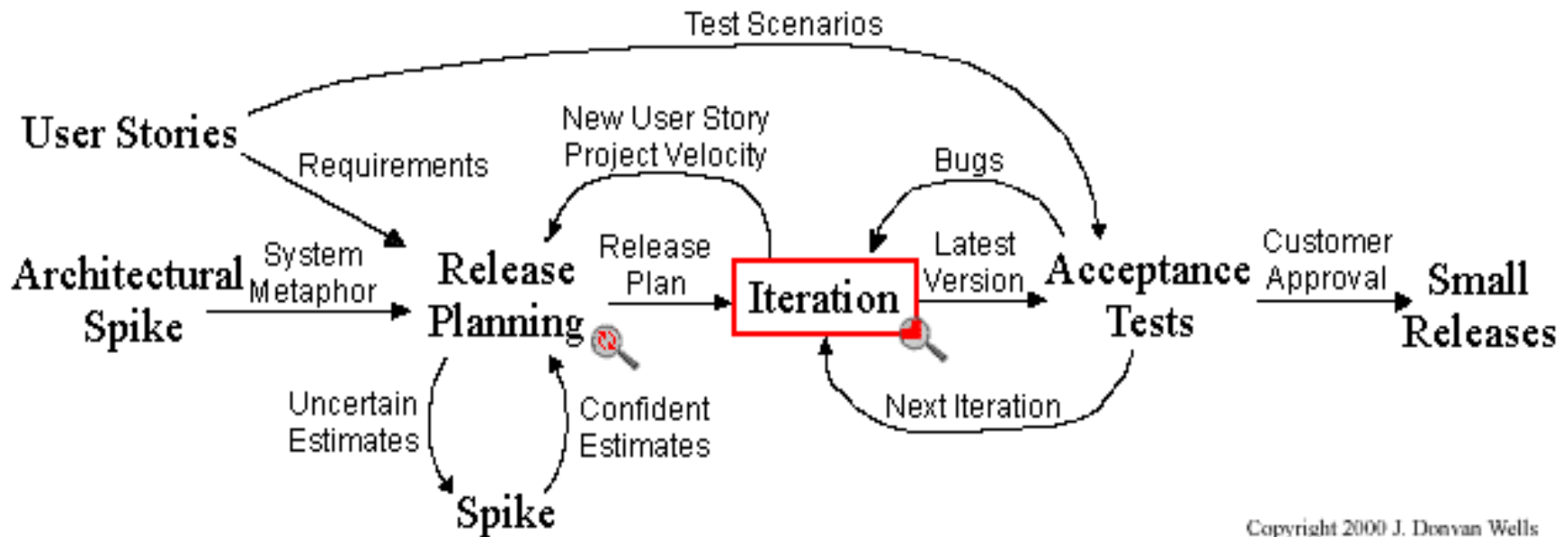
The Spiral Model



Security and Software Engineering



Extreme Programming Project



Copyright 2000 J. Donovan Wells

<http://www.extremeprogramming.org/>



General Guidelines for Developers

- Be a Minimalist / KISS
 - When possible, code should be small, simple and easy to verify.
 - Complex code increases the possibility for security vulnerabilities
- A little paranoia goes a long way
 - Ask “what if”
 - Examine consequences
 - Look for the weakest links
- Fail securely
 - Failure incorporated into design
 - No single point of failure



Secure Programming Tips - 1

- Never trust incoming data. Never.
 - Buffer overflows
 - Validate input
 - Protect settings
- Understand secure programming
 - Understand bad coding practices
 - Watch out when using dangerous languages (C, C++)
- Use code analyzers



Secure Programming Tips - 2

- Watch what you use
 - DON'T USE PRODUCTION DATA ON TEST SYSTEMS!
- Do not use more power than you actually need
 - Use administrative accounts only when necessary
- Use layers of defense
- Know when/where/how to store sensitive stuff
 - Encrypt when possible



Secure Programming Tips - 3

- Create useful logs
- Provide descriptive error messages
- Code reviews are your friends
 - They must include security reviews
- Document, document, document
- DON'T STOP LEARNING!
 - Education is a friend of security

Security Resources



- Best Practices for Secure Web Development
<http://members.rogers.com/razvan.peteanu/>
- Secure Programming for Linux and Unix HOWTO
<http://www.linuxdoc.org/HOWTO/Secure-Programs-HOWTO/>
- Security Code Guidelines
<http://java.sun.com/security/seccodeguide.html>
- The Shmoo Group - How to Write Secure Code
<http://www.shmoo.com/securecode/>
- Engineering Principles for IT Security - NIST
<http://csrc.nist.gov/publications/nistpubs/800-27/sp800-27.pdf>

Questions?



Please send all questions to:

Ron Woerner

ron_woerner@excite.com

