

Software Security: Lessons Learned in Reaching Developers and Gaining Leadership Support

Steven Lavenhar

Lavenhar_steven@bah.com

Booz Allen Hamilton

8283 Greensboro Drive

McLean VA 22102

Provided Software Security Support for Federal Agencies

- ▶ DFAS
- ▶ DOT
- ▶ HHS
- ▶ NSA
- ▶ DHS
- ▶ ED
- ▶ Interior
- ▶ Treasury
- ▶ DISA
- ▶ FDA
- ▶ IRS
- ▶ USDA
- ▶ DLA
- ▶ FEMA
- ▶ NASA
- ▶ VA
- ▶ DOD
- ▶ GAO
- ▶ NCI
- ▶ DOE
- ▶ GSA
- ▶ NIH

Software security: Why care?

- ▶ Software is ubiquitous.
- ▶ We rely on software to handle the sensitive and high-value data on which our livelihoods, privacy, and very lives depend.
- ▶ Many critical business functions in government and industry depend completely on software.
- ▶ Software—even high-consequence software—is increasingly exposed to the Internet.
 - Increased exposure makes software (and the data it handles) visible to people who never even knew it existed before.
 - Not all of those people are well-intentioned (to say the least!).



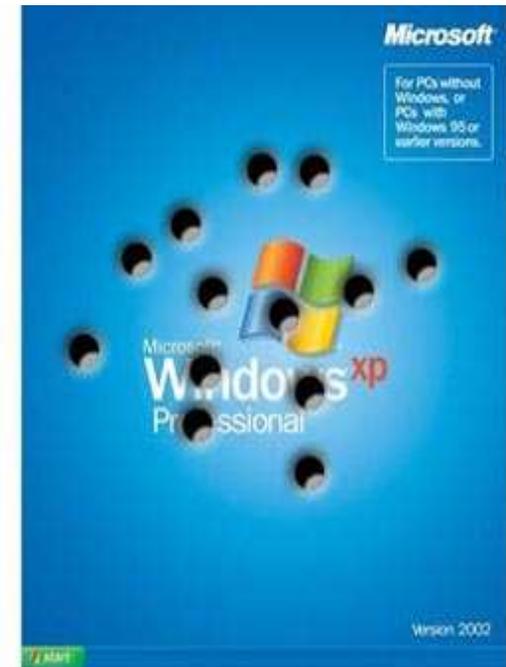
Security as a property of software

- ▶ Secure software is software that can't be intentionally forced to perform any unintended function.
- ▶ Secure software continues to operate correctly even under attack.
- ▶ Secure software can recognize attack patterns and avoid or withstand recognized attacks.
- ▶ At the whole-system level, after an attack, secure software recovers rapidly and sustains only minimal damage.



Exploitable defects in software lead to vulnerabilities

- ▶ Inherent deficiencies in the software's processing model (e.g., Web, SOA, Email) and the model's associated protocols/technologies
 - *Example:* Trust establishment in web applications is only one-way (client authenticates server)
- ▶ Shortcomings in the software's security architecture
 - *Example:* Exclusive reliance on infrastructure components to filter/block dangerous input, malicious code, etc.
- ▶ Defects in execution environment components (middleware, frameworks, operating system, etc.),
 - *Example:* Known vulnerabilities in WebLogic, J2EE, Windows XP, etc.

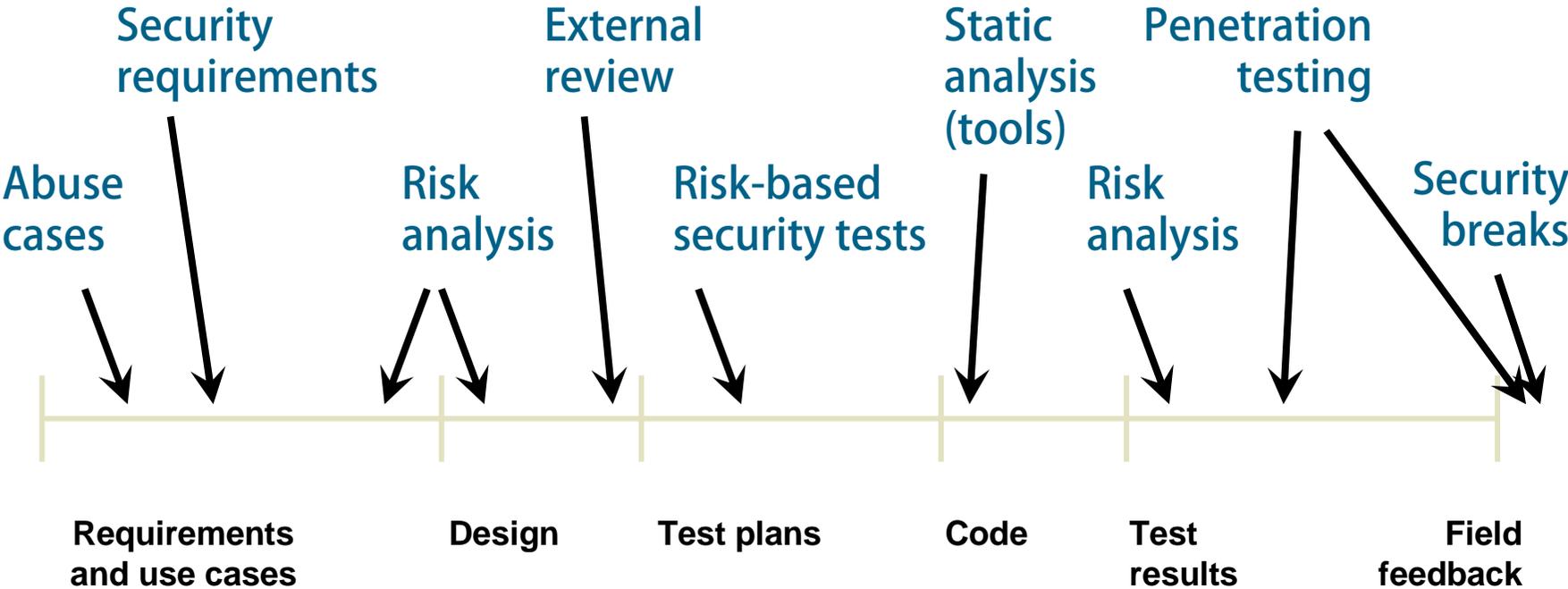
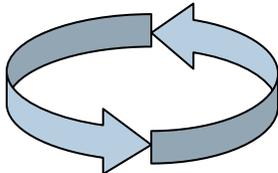


Exploitable defects cont'd

- ▶ Defects in the design or implementation of software's interfaces with environment- and application-level components
 - *Example:* Reliance on known-to-be-insecure API, RPC, or communications protocol implementations
- ▶ Defects in the design or implementation of the software's interfaces with its users (human or software process)
 - *Example:* Web application fails to establish user trustworthiness before accepting user input.
- ▶ Defects in the design or implementation of the software's processing of input
 - *Example:* C++ application does not do bounds checking on user-submitted input data before writing that data to a memory buffer.



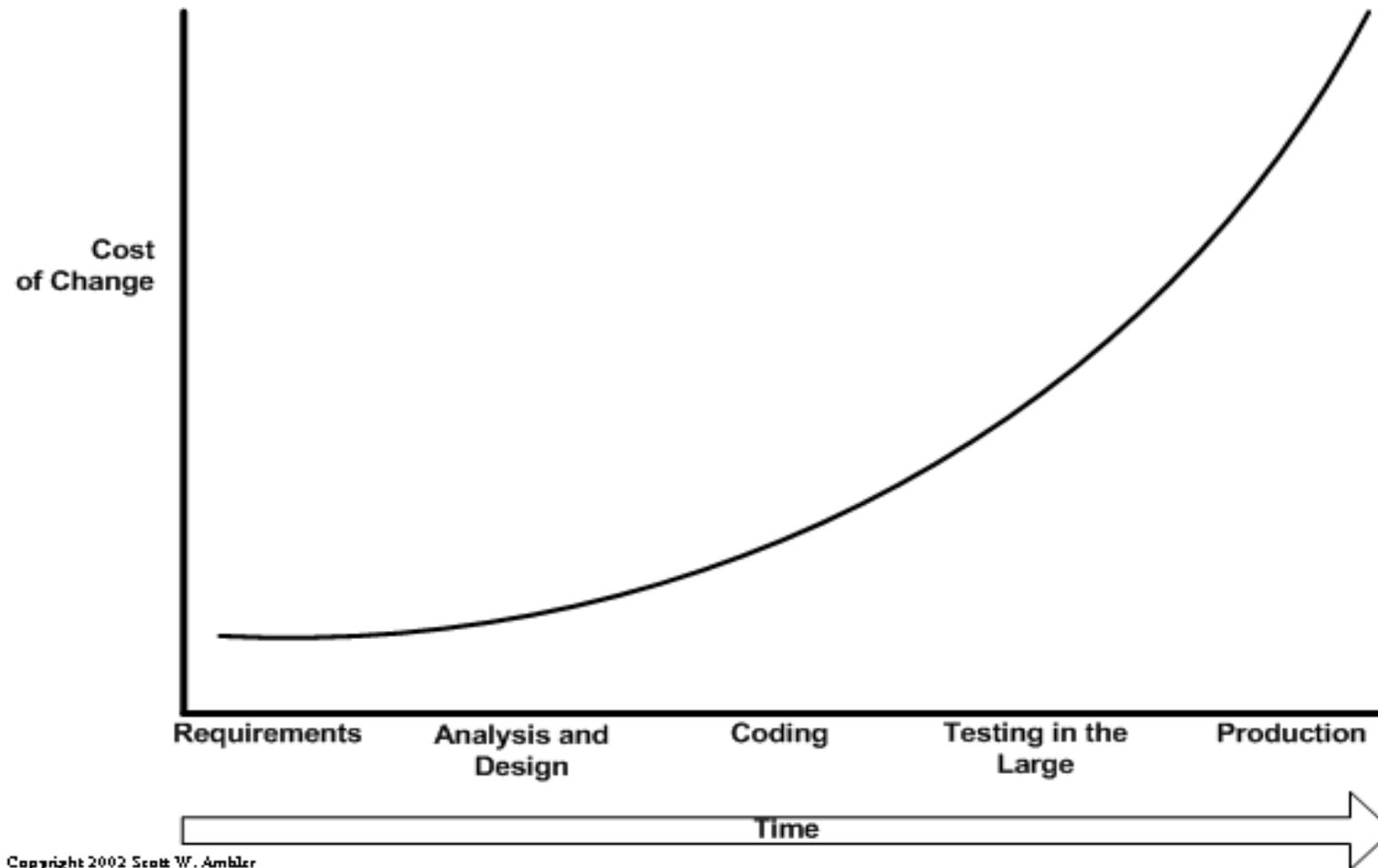
Software Security SDLC Touchpoints



Source: Gary McGraw

When to Address Software Security?

- ▶ As early as possible **AND** throughout the development lifecycle



Cost of Software Security Vulnerabilities

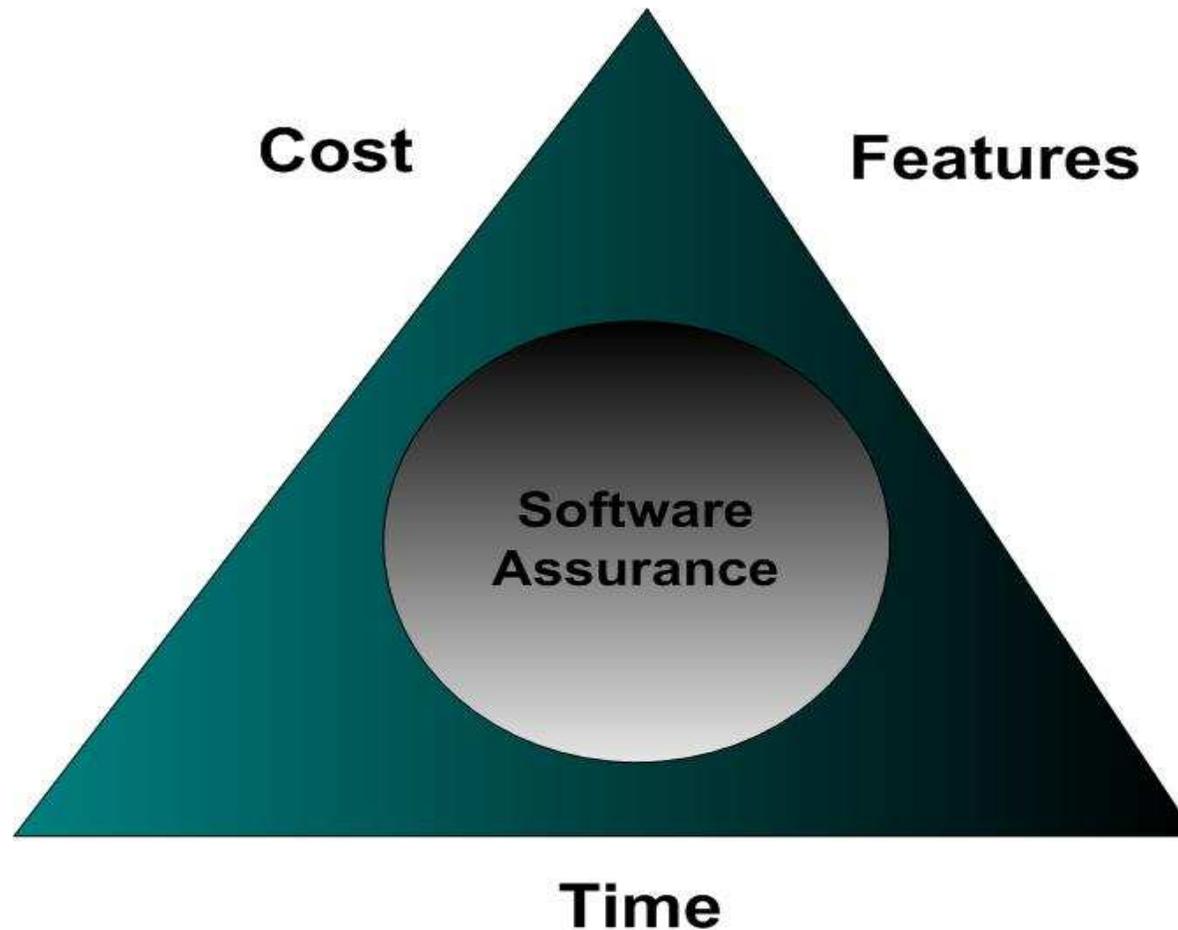
- ▶ NIST estimates costs of \$60 Billion a year due to software vulnerabilities
- ▶ Security fixes for implementation flaws typically cost \$2000-\$10,000 when done during testing phase. However, they may cost more than 5-10 times when fixed after the application has been shipped.
- ▶ The cost of fixing architectural flaws is significantly higher than fixing implementation flaws.
- ▶ Gartner Group says system downtime caused by software vulnerabilities will triple from 5% to 15% by 2008 for firms that don't take proactive security steps

So what's the Problem?

- ▶ Compliance driven culture
 - Federal laws, Executive Orders, directives, policies, standards, or regulations
 - Agency policies
 - “If it isn't in NIST Special Publication 800-53 or in a agency policy document the problem doesn't exist”
- ▶ System and network-centric focus
- ▶ Lack of software security training for developers and architects
- ▶ Federal agencies tend to be reactive rather than proactive
- ▶ The Agency's mission and business functions are almost always viewed as more important than software security
 - Lack of support from executive leadership

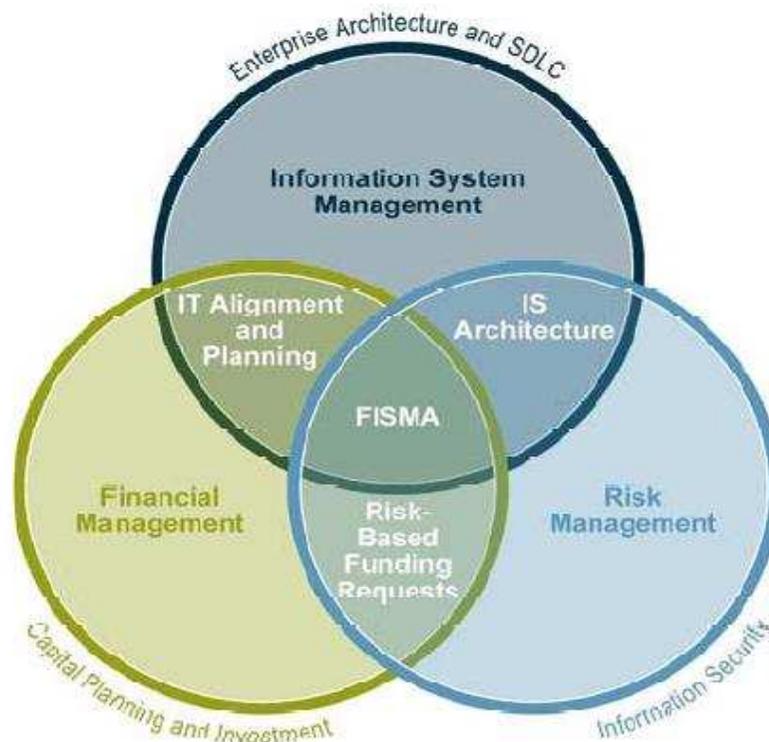
The Software Project Triangle

- ▶ Any changes you make to any side of the triangle are likely to affect software assurance



NIST Approach to Security

- ▶ A risk management approach involves continually balancing the protection of agency information and assets with the cost of security controls and mitigation strategies throughout the complete project and system development life cycle (NIST SP800-64)

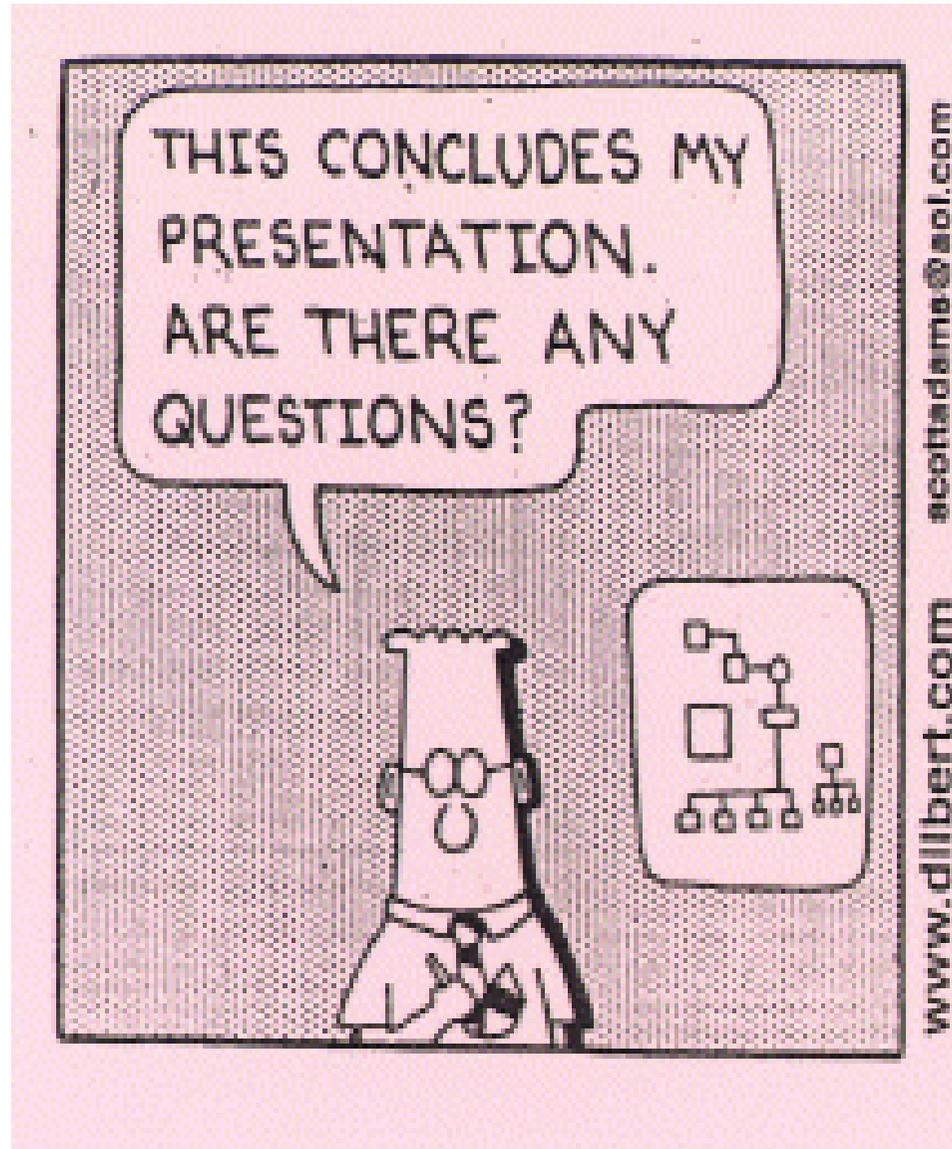


NIST SP 800-53, Control RA-5 Vulnerability Scanning

- ▶Vulnerability analysis for custom software and applications may require additional, more specialized techniques and approaches (e.g., web-based application scanners, source code reviews, source code analyzers.....)

The Solution

- ▶ Security incidents are a reality check that often force an Agency to deal with its software security issues
- ▶ Engage software security evangelists (who make slow but steady progress)
 - Introduction of software security best practices
 - Advocate for changes in policies that are software security focused
 - Advocate for executive support (but support is often from the bottom up)
- ▶ Application specific security training for developers and architects
- ▶ Present ROI data that promotes buy-in from project managers
 - Benefits accrued from improved development practices that result in better code quality
 - Positive effects on cost and schedule
- ▶ Make use of existing government resources such as the DISA Application Security And Development Security Technical Implementation Guide (STIG) and the DISA Application Security And Development Checklist



Contact Information

Steven Lavenhar
Booz Allen Hamilton
lavenhar_steven@bah.com
703-337-0392