HS SEDI

Homeland Security Systems Engineering and Development Institute

Software Security Practices

Integrating Security into the SDLC

Robert A. Martin
Sean Barnum

May 2011

Homeland Security

HS SEDI is a trademark of the U.S. Department of Homeland Security (DHS).
The HS SEDI FFRDC is managed and operated by The MITRE Corporation for DHS.

Report Documentation Page

Form Approved
OMB No. 0704-0188

1. REPORT DATE **MAY 2011**	2. REPORT TYPE	3. DATES COVERED **00-00-2011 to 00-00-2011**

4. TITLE AND SUBTITLE **Software Security Practices Integrating Security into the SDLC**	5a. CONTRACT NUMBER
	5b. GRANT NUMBER
	5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S)	5d. PROJECT NUMBER
	5e. TASK NUMBER
	5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) **The MITRE Corporation,202 Burlington Rd,Bedford,MA,01730-1420**	8. PERFORMING ORGANIZATION REPORT NUMBER
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)	10. SPONSOR/MONITOR’S ACRONYM(S)
	11. SPONSOR/MONITOR’S REPORT NUMBER(S)

12. DISTRIBUTION/AVAILABILITY STATEMENT

Approved for public release; distribution unlimited

13. SUPPLEMENTARY NOTES

Presented at the 23rd Systems and Software Technology Conference (SSTC), 16-19 May 2011, Salt Lake City, UT. Sponsored in part by the USAF. U.S. Government or Federal Rights License

14. ABSTRACT

15. SUBJECT TERMS

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT **unclassified**	b. ABSTRACT **unclassified**	c. THIS PAGE **unclassified**	**Same as Report (SAR)**	**19**	

Agenda

8:00-8:45am	Software Security Knowledge about Applications Weaknesses
9:00-9:45am	Software Security Knowledge about Attack Patterns Against Applications
	Training in Software Security
10:15-11:00am	Software Security Practice
11:15-12:00am	Supporting Capabilities
	Assurance Cases
	Secure Development & Secure Operations

Planning for Software Security

- **Some questions to aid in understanding security risks to achieving project goals and objectives:**
  - What is the value we must protect?
  - To sustain this value, which assets must be protected? Why must they be protected? What happens if they’re not protected?
  - What potential adverse conditions and consequences must be prevented and managed? At what cost? How much disruption can we stand before we take action?
  - How do we determine and effectively manage residual risk (the risk remaining after mitigation actions are taken)?
  - How do we integrate our answers to these questions into an effective, implementable, enforceable security strategy and plan?
- **Help you determine how much to invest, where to invest, and how fast to invest in an effort to mitigate software security risk.**

Influencing the Security Properties of Software

- **Balance between engaging in defensive action and thinking like an attacker**

- **Primary perspective is that of defender**
  - Build in security features to make software resilient to attack
  - Minimize weaknesses that may lead to vulnerability

- **Balancing perspective is that of the attacker**
  - Strive to understand the exact nature of the threat that the software is likely to face so as to focus defensive efforts on areas of highest risk.

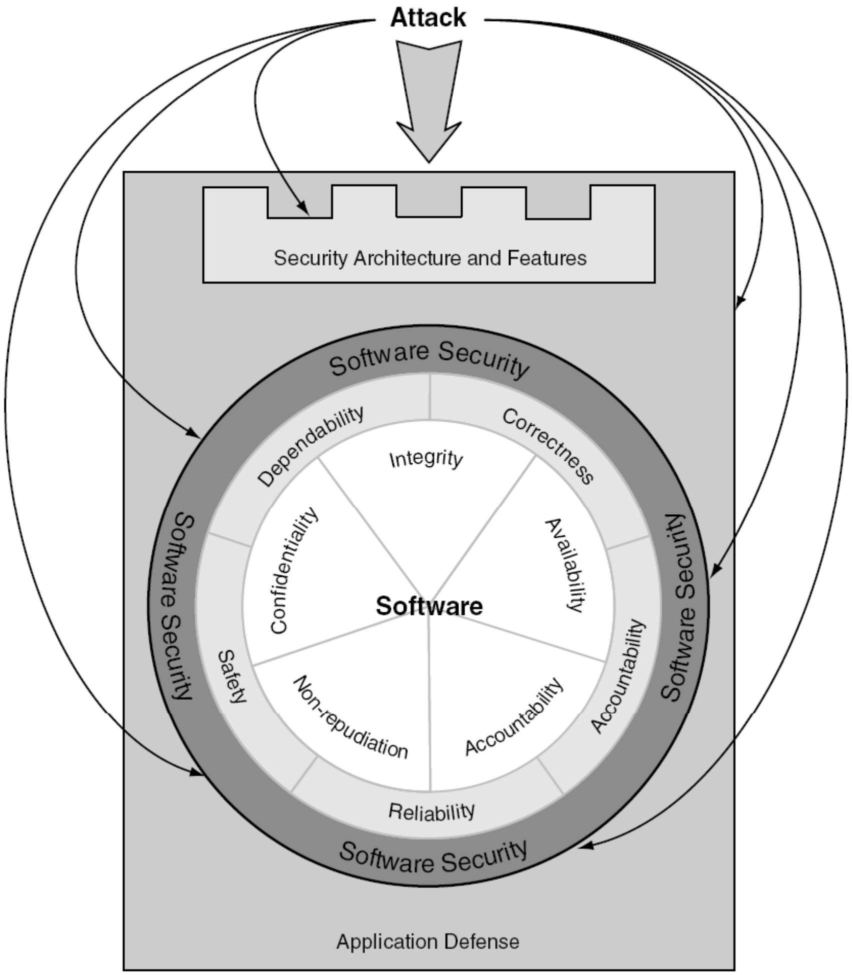
- **These two perspectives, working in combination, guide the actions taken to make software more secure.**

Addressing the Expected & Unexpected: Avoiding, Removing, and Mitigating Weaknesses – Software Security

- **“software security” focuses on preventing weaknesses from entering the software in the first place or, if that is unavoidable, at least removing them as early in the life cycle as possible and before the software is deployed**

- **Build Security In!!**

- **A wide variety of security-focused practices are available to software project managers and their development teams that can be seamlessly integrated throughout any typical software engineering SDLC**



Integrating Security into a typical software development lifecycle (SDLC) is evolutionary not revolutionary

It is fundamentally an extension of good quality practices

Traditional Quality Assurance

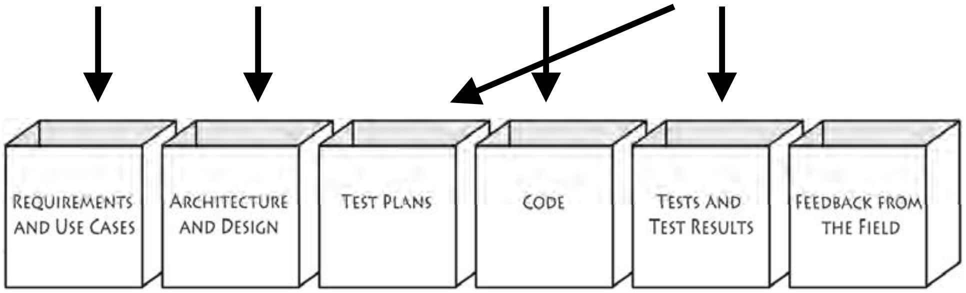
- Requirements Reviews
- Design Reviews
- Code Reviews
- Traditional Testing

Requirements Review

Design Review

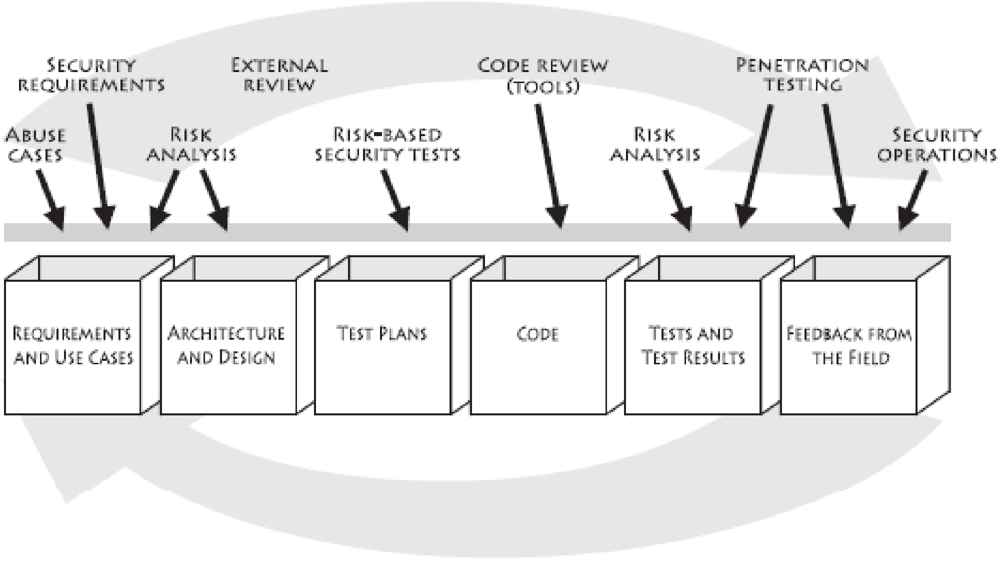
Code Review

Test Strategy, Planning & Execution



Extending Traditional QA to Include Security

- Security Requirements Capture and Analysis including Abuse Cases
- Architectural Risk Analysis
- Secure Code Review
- Risk-based Security Testing
- Penetration Testing



What New Dimensions does Security Bring?

- **Don’t stop what you are doing, just build on it**
- **Evidence that software does what it is supposed to do and nothing else**
- **Intentional vs Unintentional problems**
- **Testing Inside-Out not just Outside-In**
- **Recognize the attacker’s perspective**
  - Think like the bad guys
- **Risk-based approach**
  - Software will never be perfect
  - Valid and valuable for QA
  - Crucial for security

Software Security Critical Lessons

- **Software security is more than a set of security functions**
  - Not silver-bullet security mechanisms
  - Not application of very simple tools
- **Non-functional aspects of design are essential**
- **Security is an emergent property of the entire system (just like quality)**
- **Breaking stuff is important**
- **To end up with secure software, deep integration with the SDLC is necessary**

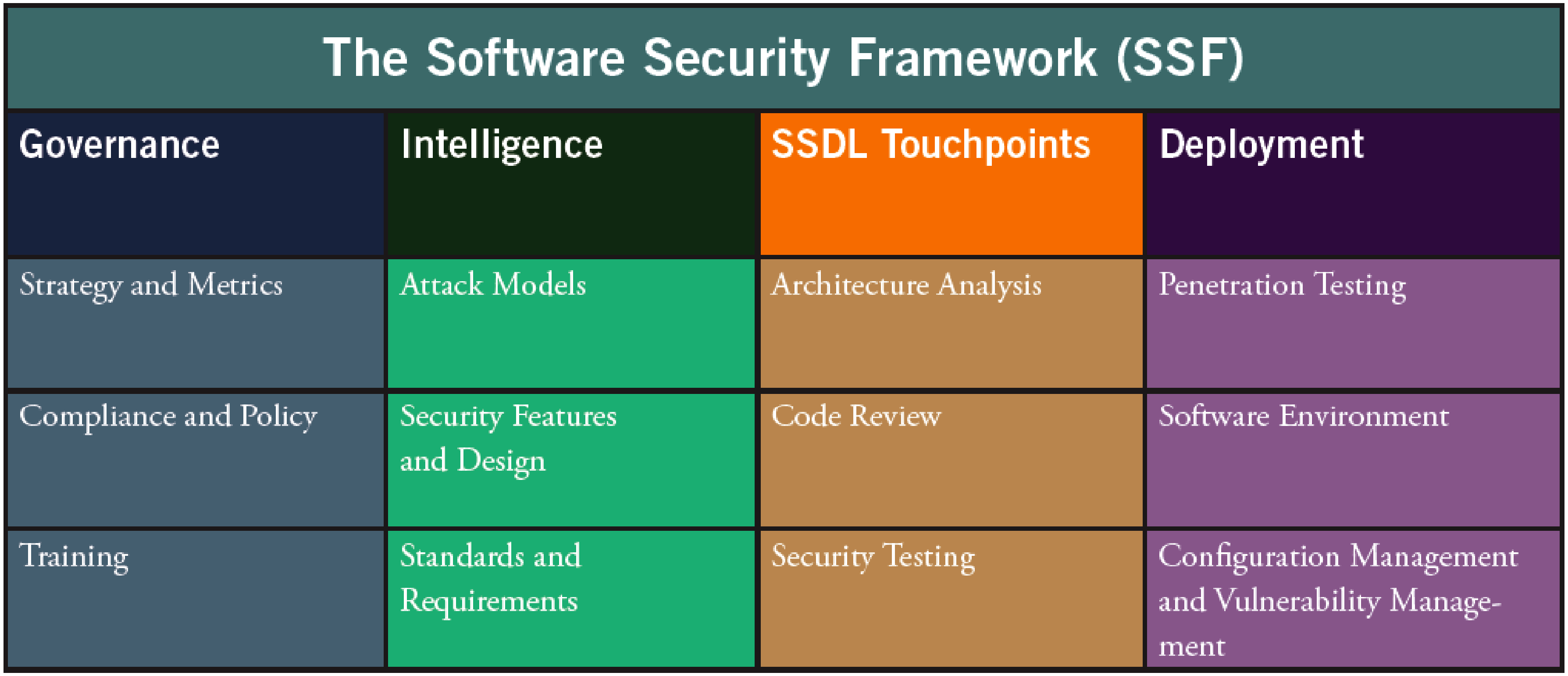
Bottom Up Software Security Actions

- **A few relatively simple things can make a tangible difference and can help you get started with software security**
- **Build checklists and use them**
  - Sun's Security at a Glance (SAG) checklist http://www.securecoding.org/companion/checklists/SAG/
- **Begin to develop a resource set (e.g., portal)**
- **Start small with simple architectural risk analyses**
- **Don't forget to include business-case justifications**
- **Use code scanning tools**

Top-Down Software Security Actions

- **Think of the problem as an evolutionary approach**
- **Chart out a strategic course of action to get where you want to be**
  - Have a gap analysis performed
  - Make achievable, realistic milestones
  - Think about metrics for success
- **Use outside help as you need it**

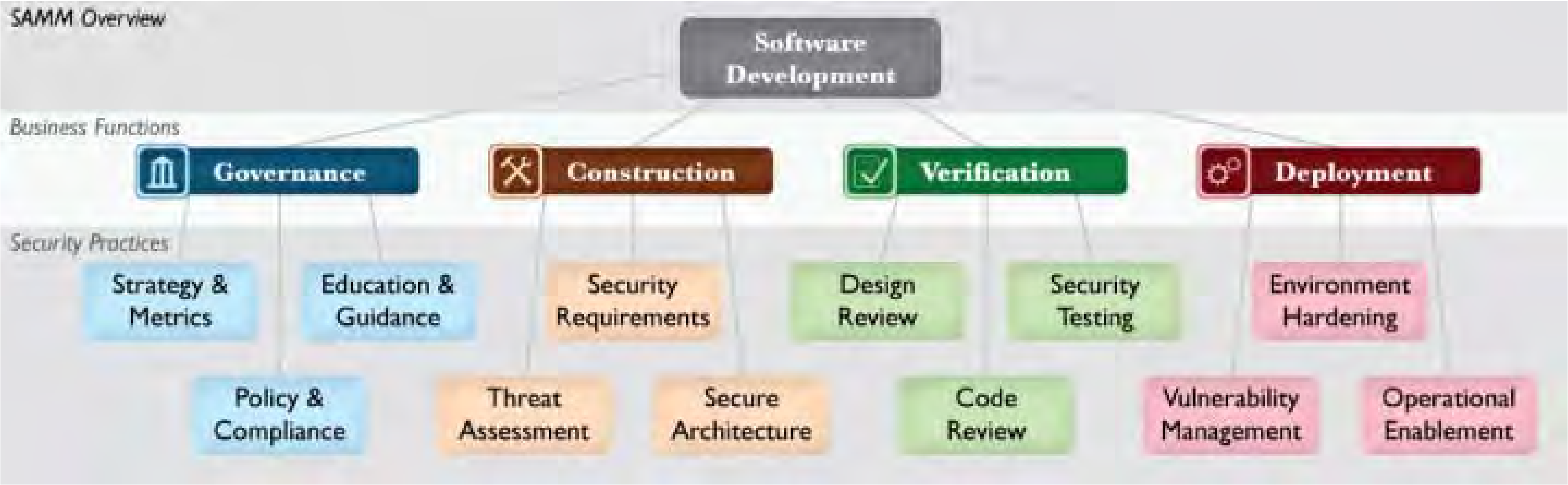
BSIMM Software Security Framework



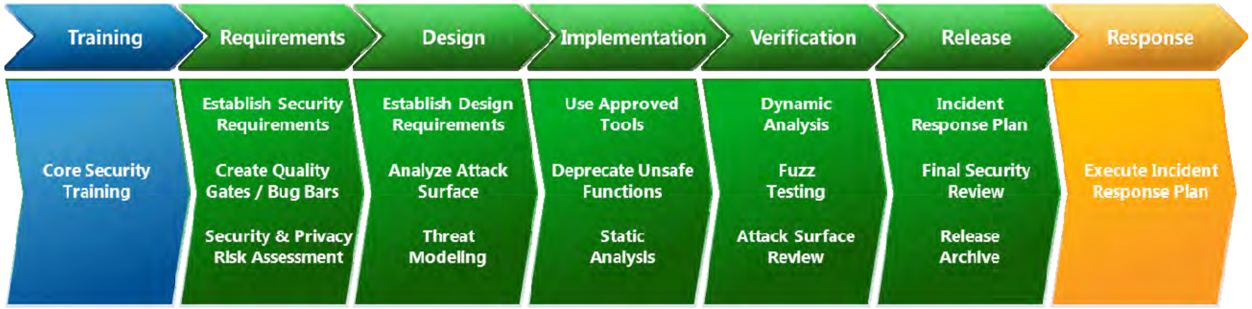
The Software Security Framework (SSF)

Governance	Intelligence	SSDL Touchpoints	Deployment
Strategy and Metrics	Attack Models	Architecture Analysis	Penetration Testing
Compliance and Policy	Security Features and Design	Code Review	Software Environment
Training	Standards and Requirements	Security Testing	Configuration Management and Vulnerability Management

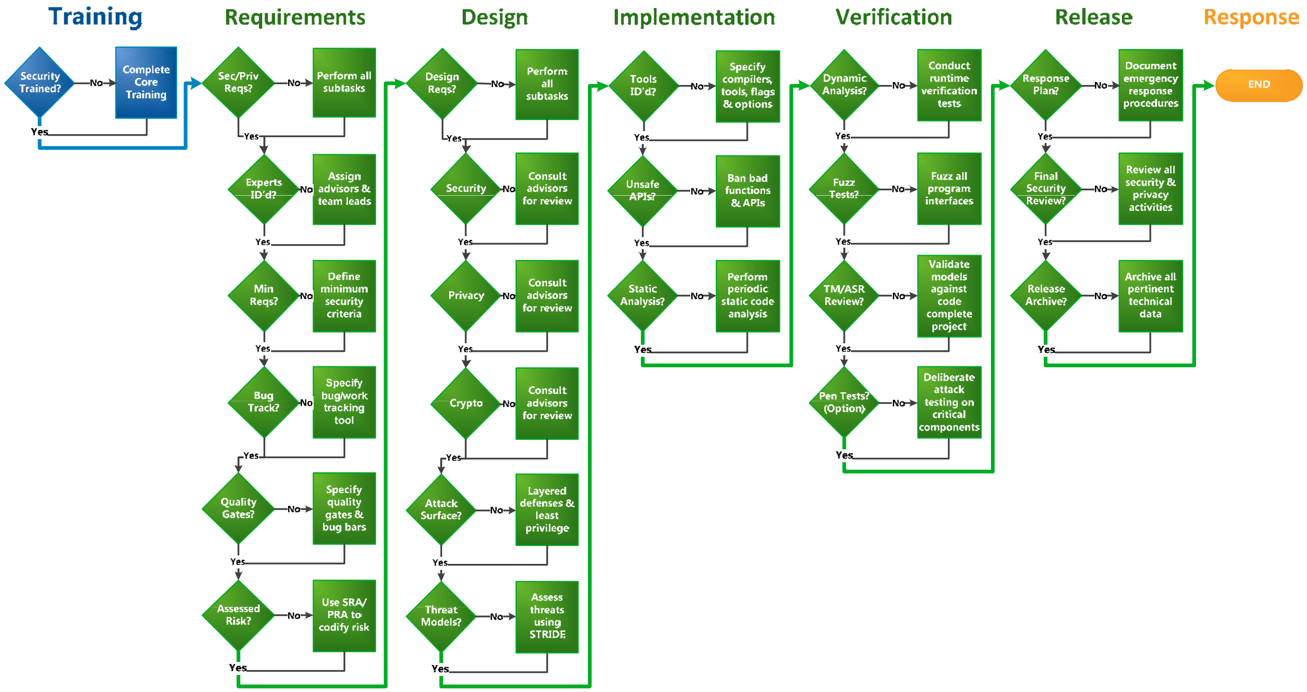
OpenSAMM


SAMM Overview
Software Development
Business Functions
Governance
Construction
Verification
Deployment
Security Practices
Strategy & Metrics
Education & Guidance
Policy & Compliance
Security Requirements
Threat Assessment
Secure Architecture
Design Review
Security Testing
Code Review
Environment Hardening
Vulnerability Management
Operational Enablement

Microsoft Secure Development Lifecycle



Training	Requirements	Design	Implementation	Verification	Release	Response
Core Security Training	Establish Security Requirements Create Quality Gates / Bug Bars Security & Privacy Risk Assessment	Establish Design Requirements Analyze Attack Surface Threat Modeling	Use Approved Tools Deprecate Unsafe Functions Static Analysis	Dynamic Analysis Fuzz Testing Attack Surface Review	Incident Response Plan Final Security Review Release Archive	Execute Incident Response Plan



Training

Security Trained? —No→ Complete Core Training; Yes

Requirements

Sec/Priv Reqs? —No→ Perform all subtasks; Yes
Experts ID'd? —No→ Assign advisors & team leads; Yes
Min Reqs? —No→ Define minimum security criteria; Yes
Bug Track? —No→ Specify bug/work tracking tool; Yes
Quality Gates? —No→ Specify quality gates & bug bars; Yes
Assessed Risk? —No→ Use SRA/ PRA to codify risk; Yes

Design

Design Reqs? —No→ Perform all subtasks; Yes
Security —No→ Consult advisors for review; Yes
Privacy —No→ Consult advisors for review; Yes
Crypto —No→ Consult advisors for review; Yes
Attack Surface? —No→ Layered defenses & least privilege; Yes
Threat Models? —No→ Assess threats using STRIDE; Yes

Implementation

Tools ID'd? —No→ Specify compilers, tools, flags & options; Yes
Unsafe APIs? —No→ Ban bad functions & APIs; Yes
Static Analysis? —No→ Perform periodic static code analysis; Yes

Verification

Dynamic Analysis? —No→ Conduct runtime verification tests; Yes
Fuzz Tests? —No→ Fuzz all program interfaces; Yes
TM/ASR Review? —No→ Validate models against code complete project; Yes
Pen Tests? (Option) —No→ Deliberate attack testing on critical components; Yes

Release

Response Plan? —No→ Document emergency response procedures; Yes
Final Security Review? —No→ Review all security & privacy activities; Yes
Release Archive? —No→ Archive all pertinent technical data; Yes

Response

END

Best Practices Reprise

- **These best practices should be applied throughout the lifecycle**
- **Tendency is to "start right" (penetration testing) and declare victory**
  - Not cost effective
  - Hard to fix problems
- **Start as far to the left as possible**

- **Abuse cases**
- **Security requirements analysis**
- **Architectural risk analysis**
- **Risk analysis at design**
- **External review**
- **Test planning based on risks**
- **Code review with static analysis tools**
- **Security testing (malicious tests)**

Summary

- Evolutionary not Revolutionary
- Security is an extension of Quality Assurance
- Requires more Inside-Out analysis
- Think like an attacker
- Risk Management is essential
- Think bottom-up (tactically) and top-down (strategically)
- Understand your context to know where you want to go
- Understand your current state to know how to get there
- Build and follow a roadmap for gradual evolution

Resources

- **Resources available with practice specifications**
  - Build Security In website (DHS)
    - https://buildsecurityin.us-cert.gov/daisy/bsi/home.html/
  - Software Assurance Self Assessment (BSIMM, SAFECode, MS SDL, etc.)
    - https://buildsecurityin.us-cert.gov/swa/proself_assm.html
  - Software Security Engineering: A Guide for Project Managers (Book)
    - http://www.softwaresecurityengineering.com/
  - Open Web Application Security Project (OWASP)
    - http://www.owasp.org

HS SEDI
Homeland Security Systems Engineering and Development Institute

Questions?

Sean Barnum
MITRE
sbarnum@mitre.org