CERT

Software Security Engineering Lecture 2

Nancy R. Mead, SEI
nrm@sei.cmu.edu

Outline

Software assurance practices

Software assurance lifecycle models

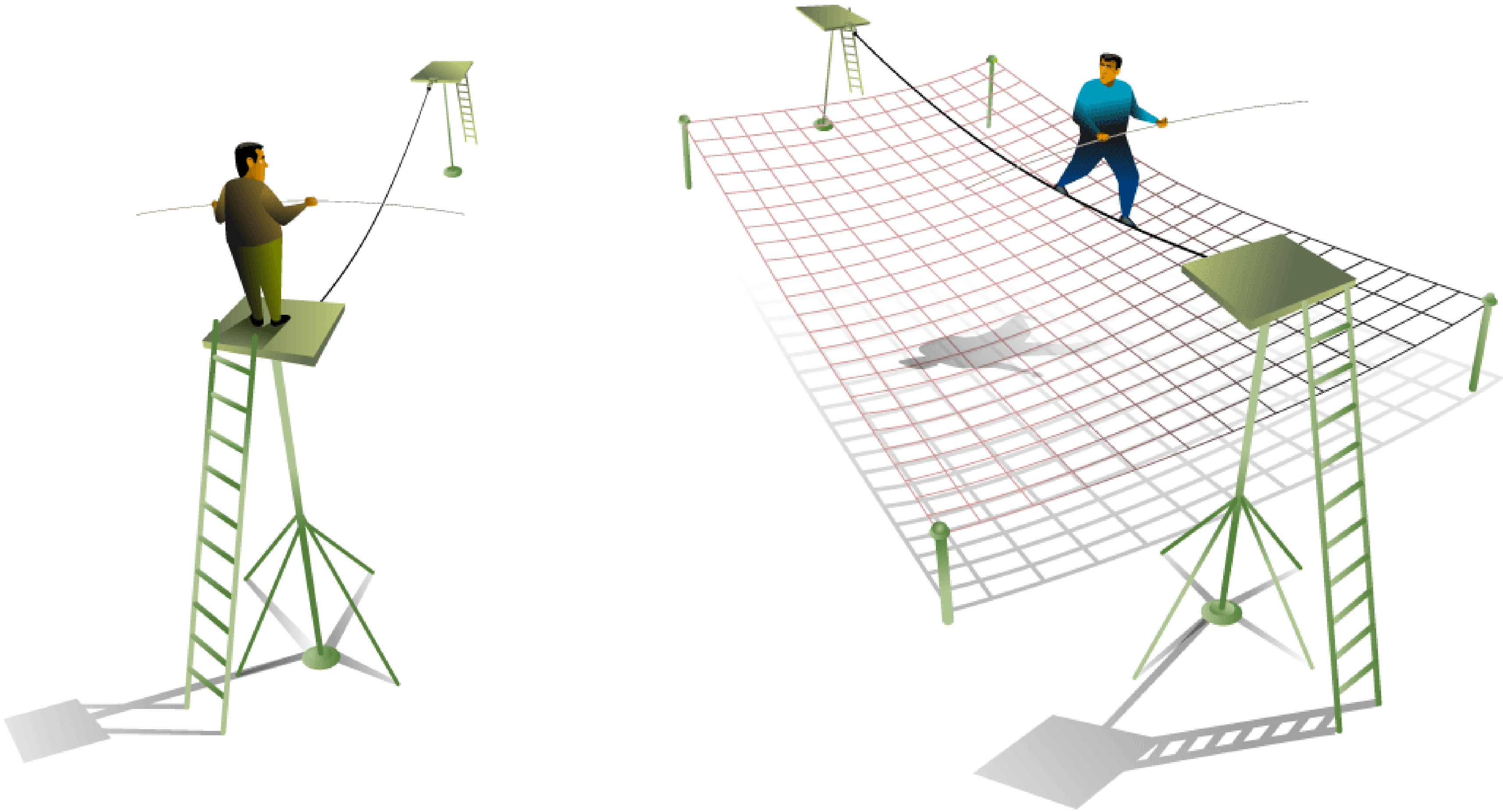
Software assurance maturity models

Software Assurance Practices

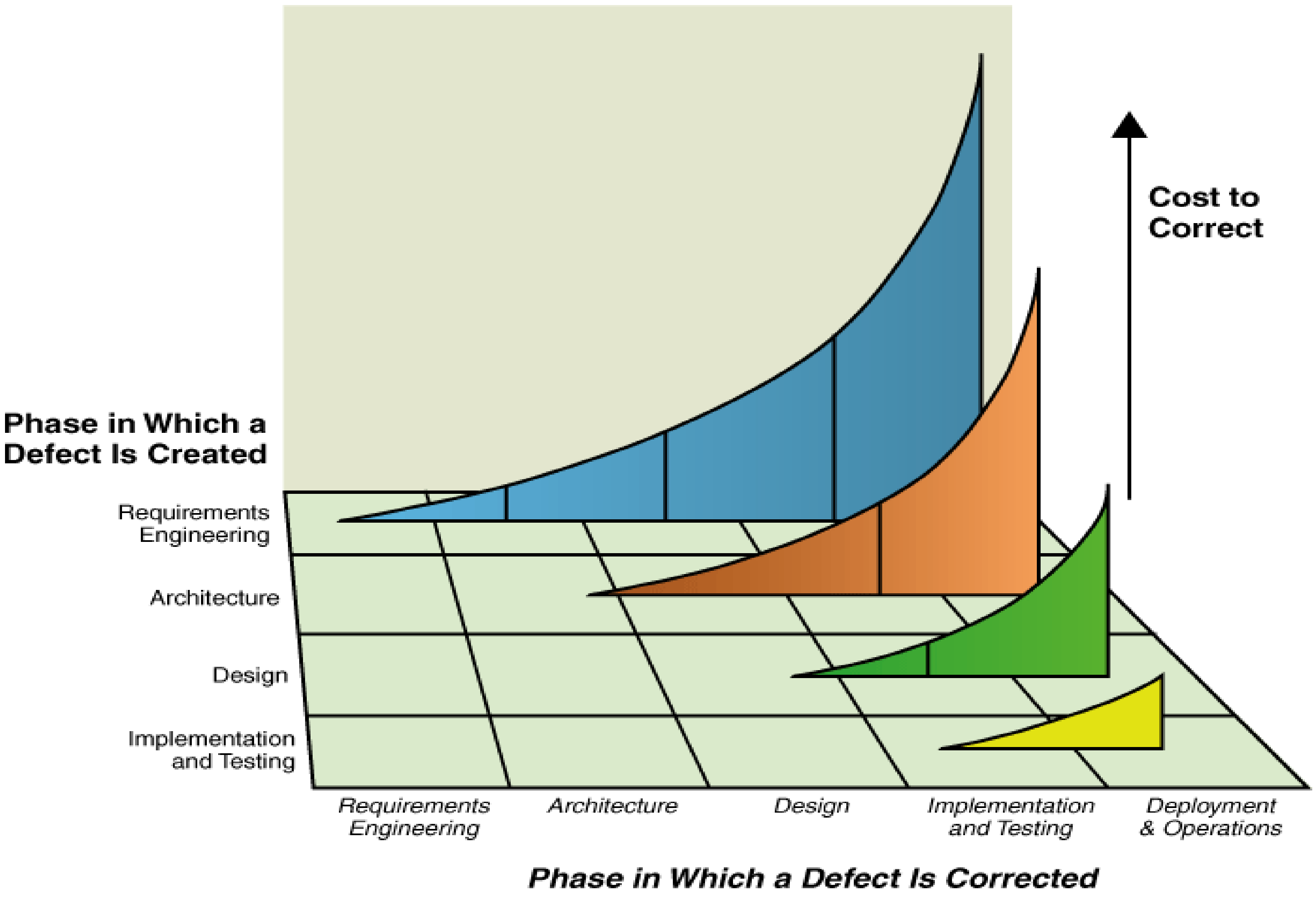
Security Perspectives

http://security.gloriad.org/blog/2007/10/21/traditional-thinking/

So What Should We Do?



Understand the Cost of Correcting Software Defects



- McConnell, Steve. “Software Quality at Top Speed.” August 1996. http://www.stevemcconnell.com

Example Security Practices - 1

- Project management
  - Enterprise software security framework
  - Security development life cycle
  - Risk management & ongoing assessment
- Full life cycle
  - Attack patterns: a structured representation for how attackers think
  - Assurance cases: demonstration that a system satisfies its security properties
- Requirements engineering
  - Misuse/abuse cases: anticipate abnormal behavior

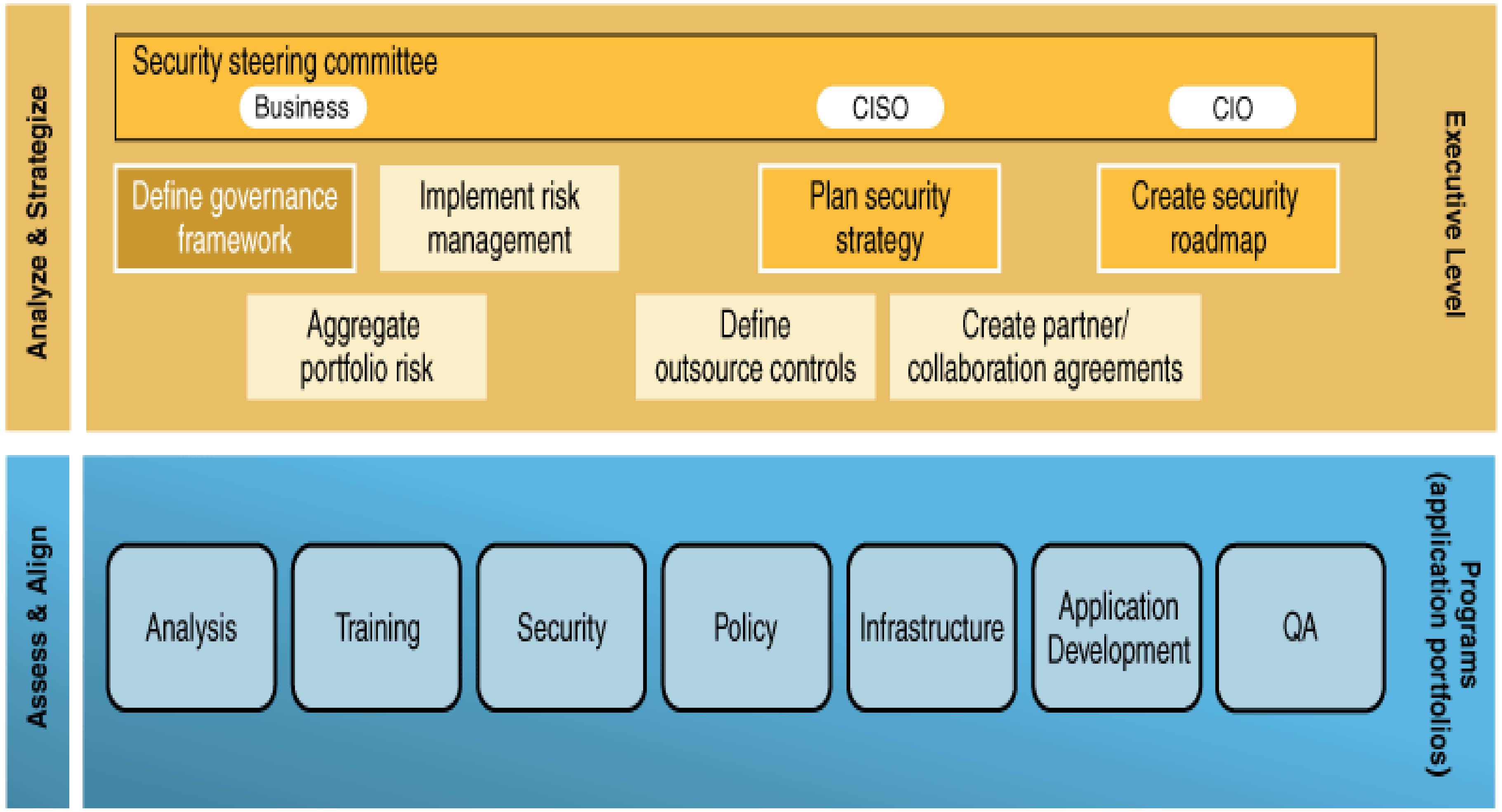
Example Security Practices - 2

- Architecture & design
  - Architectural risk analysis
- Code & test
  - Secure code reviews
  - White box, black box, & penetration testing

CERT

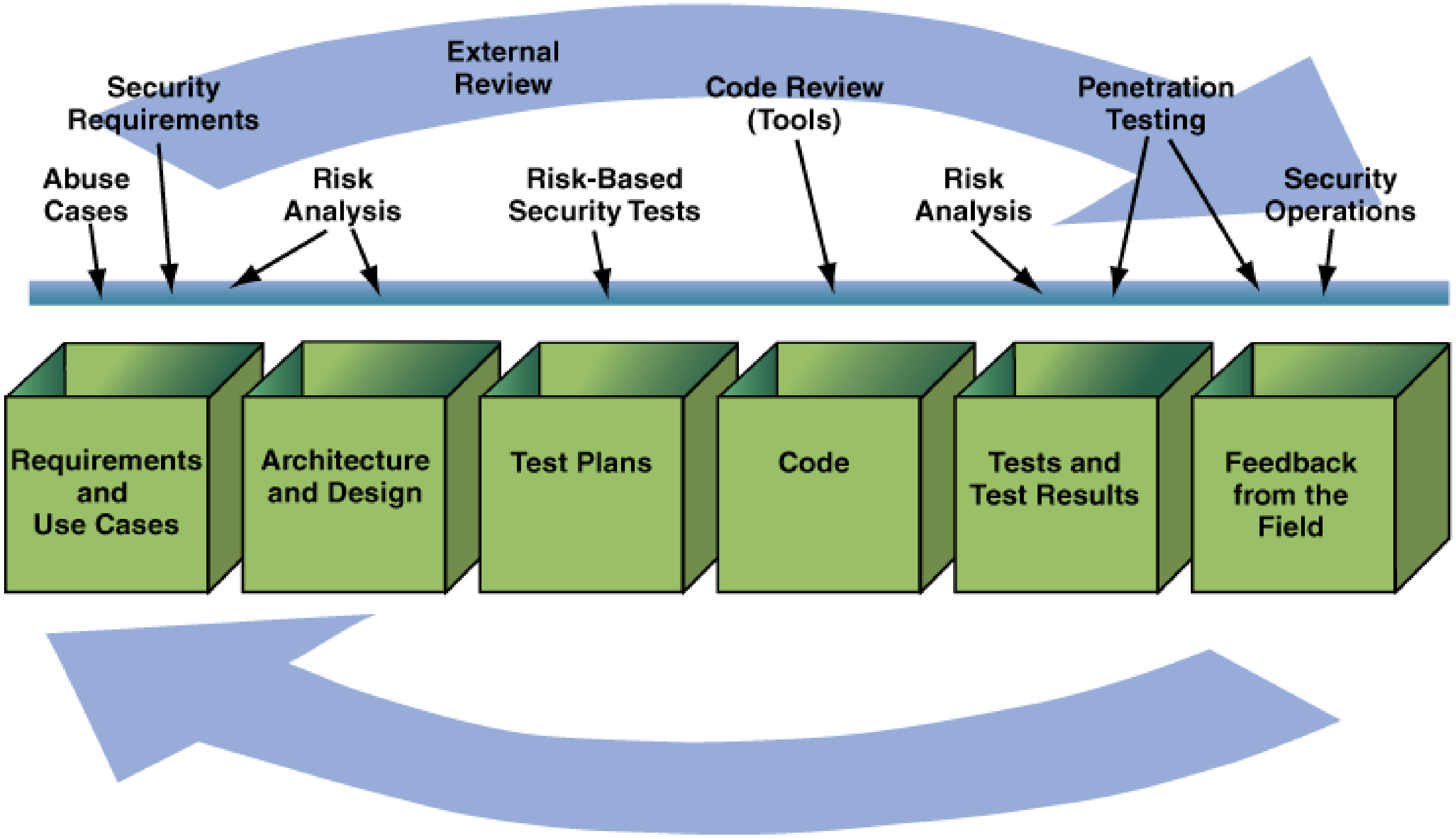
Software Assurance Lifecycle Models

Enterprise Software Security Framework



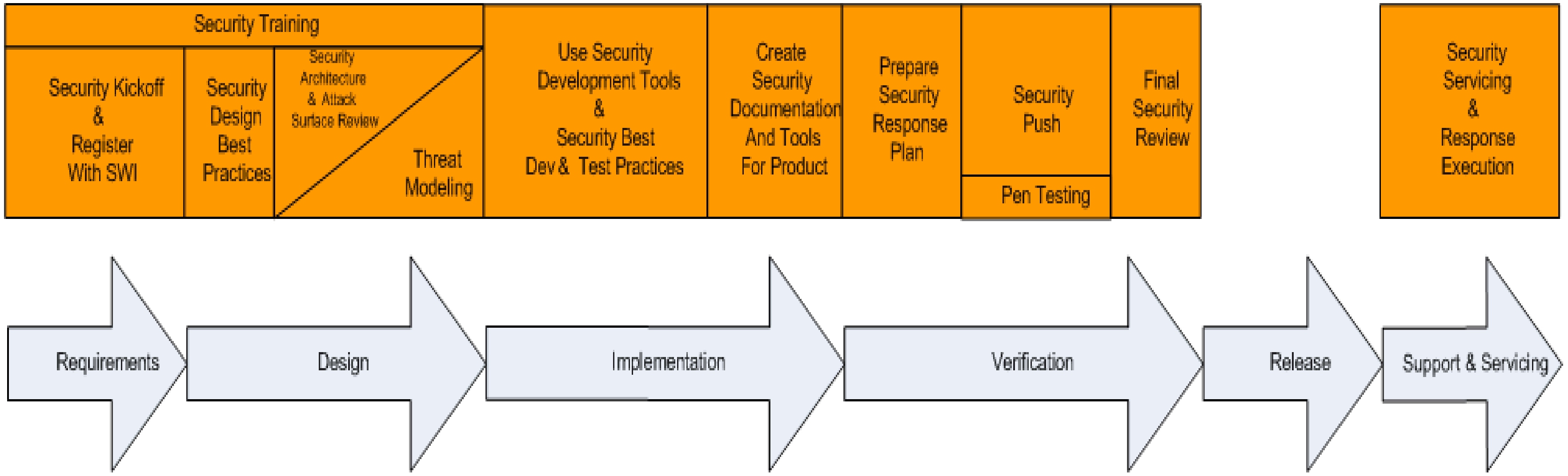
Steven, John. "Adopting an Enterprise Software Security Framework." *IEEE Security & Privacy 4*, 2 (March/April 2006): 84–87. https://buildsecurityin.us-cert.gov/daisy/bsi/resources/published/series/bsi-ieee/568.html

SDLC With Defined Security Touchpoints



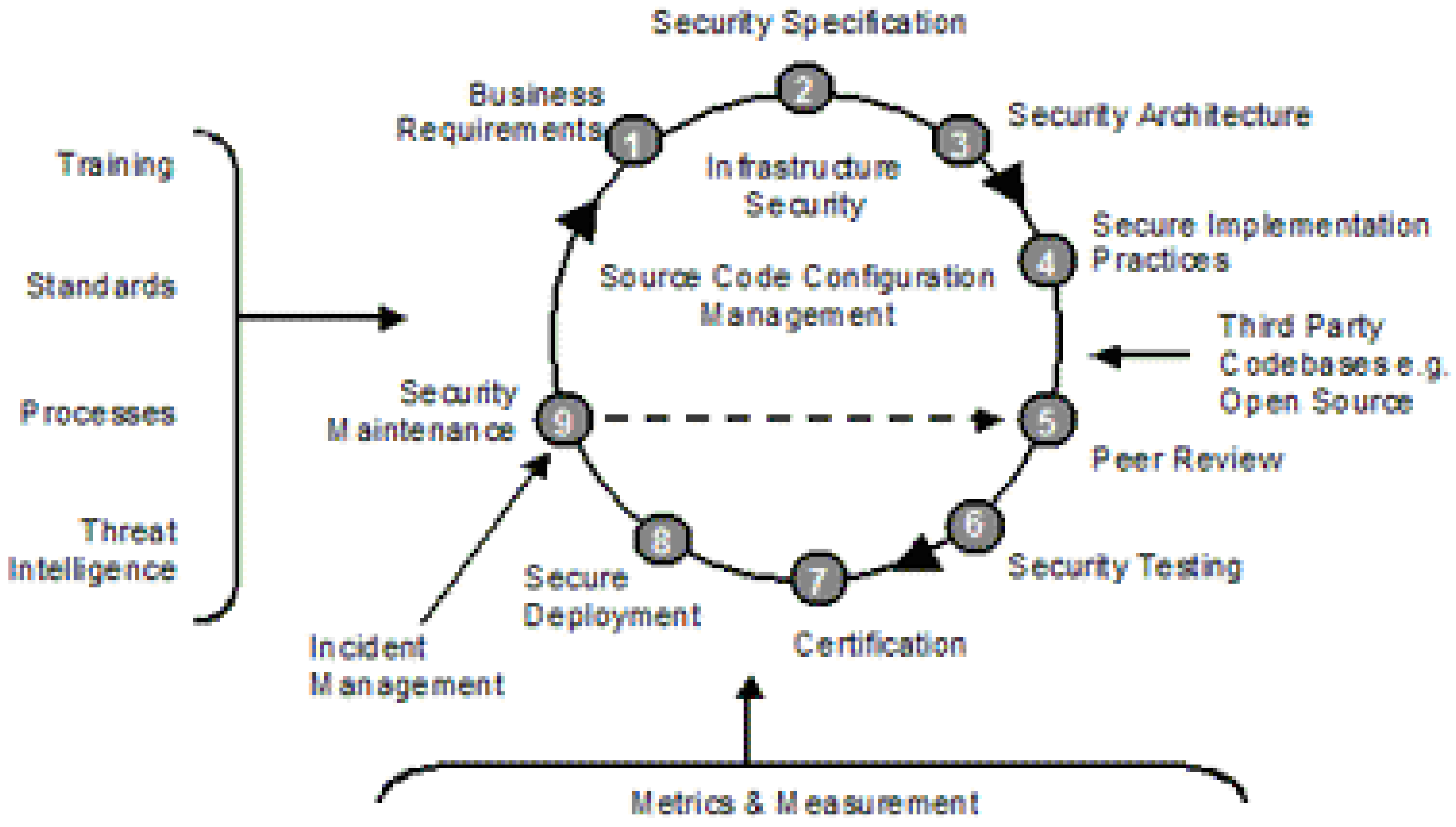
SDLC: Software Development Life Cycle
McGraw, Gary. *Software Security: Building Security In*. Boston, MA: Addison-Wesley Professional, 2006.

Microsoft's Security Development Lifecycle


Security Training
Security Kickoff & Register With SWI
Security Design Best Practices
Security Architecture & Attack Surface Review
Threat Modeling
Use Security Development Tools & Security Best Dev & Test Practices
Create Security Documentation And Tools For Product
Prepare Security Response Plan
Security Push
Pen Testing
Final Security Review
Security Servicing & Response Execution
Requirements
Design
Implementation
Verification
Release
Support & Servicing

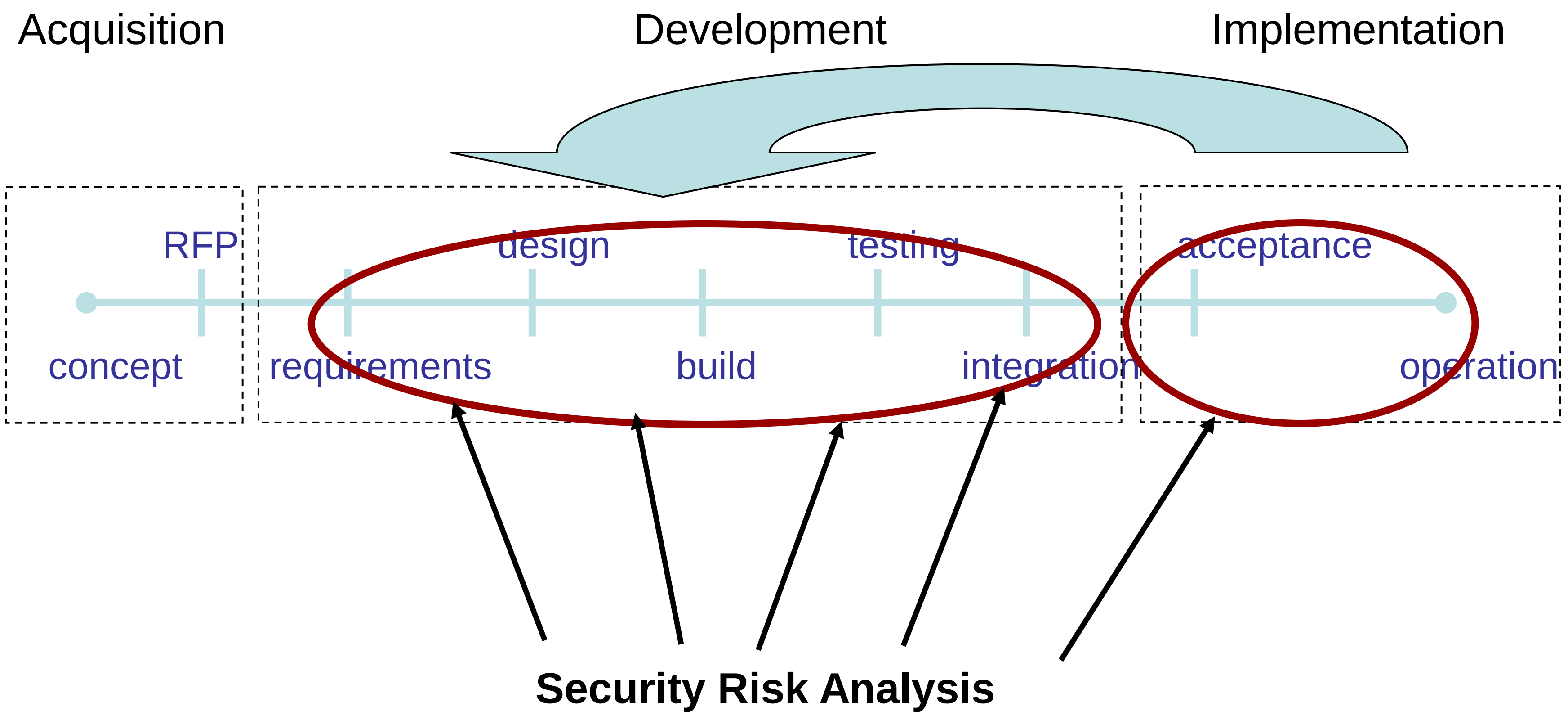
http://msdn2.microsoft.com/en-us/library/ms995349.aspx

Assurent Software Security Lifecycle



http://www.assurent.com/index.php?id=59

Assess Security Risk Across the SDLC



Attack Patterns

- Blueprint for creating an attack (like a sewing pattern)
- Consists of:
  - Attack prerequisites
  - Attack description
  - Related vulnerabilities
  - Method of attack
  - Skill & resources required to execute attack
  - Applicable contexts
  - Prevention & mitigation strategies

Consult CAPEC: Common Attack Pattern Enumeration and Classification http://capec.mitre.org/

Assurance Cases

- Applicable during all phases of software development
- Similar to a legal case
- Presents arguments showing how a top-level claim is supported by evidence
  - The system is acceptably secure
  - The system has none of the common coding defects that lead to security vulnerabilities
- Considers people, process, and technology

Misuse/Abuse Cases

- Document *a priori* how software should react to illegitimate use (can’ts and won’ts)
  - Brainstorm with designers and software security experts
    - How does the software distinguish between good and bad input?
    - Between legitimate application vs. rogue application requests?
    - How can an attacker disrupt software communication interfaces?
    - Does the database server assume that the client manages all data access permissions?
- Ask:
  - What assumptions are implicit in our system?
  - What things make our assumptions false?
  - What are some candidate attacks (consult attack patterns)?
- Strike a balance between cost and value
  - Prioritize which cases to develop
  - Risk analysis helps guide case selection

Architecture & Design

- Not the same as security architecture
  - architecture of security components (firewalls, IDS, other sensors, network monitoring points, etc.)
- Architectural Risk Analysis
  - software characterization
  - threat analysis
  - architectural vulnerability assessment
  - risk likelihood determination
  - risk impact determination
  - risk mitigation planning

Secure Code Review/Scanning

- Adopt a secure coding standard
  - Validate input
  - Perform bounds checking (buffer overflows)
  - Check for conditions that could lead to exceptions
  - Base access decisions on permission, not exclusion (default deny)
  - Enforce the principle of least privilege for processes
    - Time out elevated privileges
  - Sanitize data sent to other systems
  - Guard against race conditions (infinite loops, deadlocks, resource collisions)
  - Review code against attack patterns & misuse/abuse cases
- Conduct structured code inspections & peer review of source code
- Use static source code analysis tools

Security Testing - 1

- Test approach & selection determined based on risk analysis
  - Use attack patterns & abuse cases
- Emphasizes what an application should *not* do
  - "Unauthorized users should not be able to access data."
    - — Validate least privilege
    - — Time-limited escalation of privilege
    - — Disable account after x unsuccessful login attempts

Security Testing - 2

- White box testing
  - validate design decisions & assumptions
  - analyze data, control, information flows; coding practices; exception & error handling
- Black box testing
  - focus on externally visible behavior
  - examine requirements, protocols, interfaces, attempted attacks
  - vulnerability scanning is one example
- Penetration testing (revised)
  - final production environment; final configuration
  - structured to demonstrate impact of likely risks

Software Assurance Maturity Models and Frameworks

Developed by Dan Reddy EMC-2

Product Security Office: Delivers Product Security From Concept to Customer

Concept

Security Development Lifecycle

Security Certifications

Vulnerability Response

Software Supply Chain Risk Management

Customer

Cross Industry Involvement

SAFECode Software Assurance Forum for Excellence in Code Driving Security and Integrity

Founding member '07

BSIMM

"… The data show that EMC's Product Security Office practices have improved greatly over time and currently rank among the most advanced."

THE Open GROUP

***Trusted Technology Forum**: Building Industry Standard for Supply Chain*

BSIMM3: The Building Security In Maturity Model

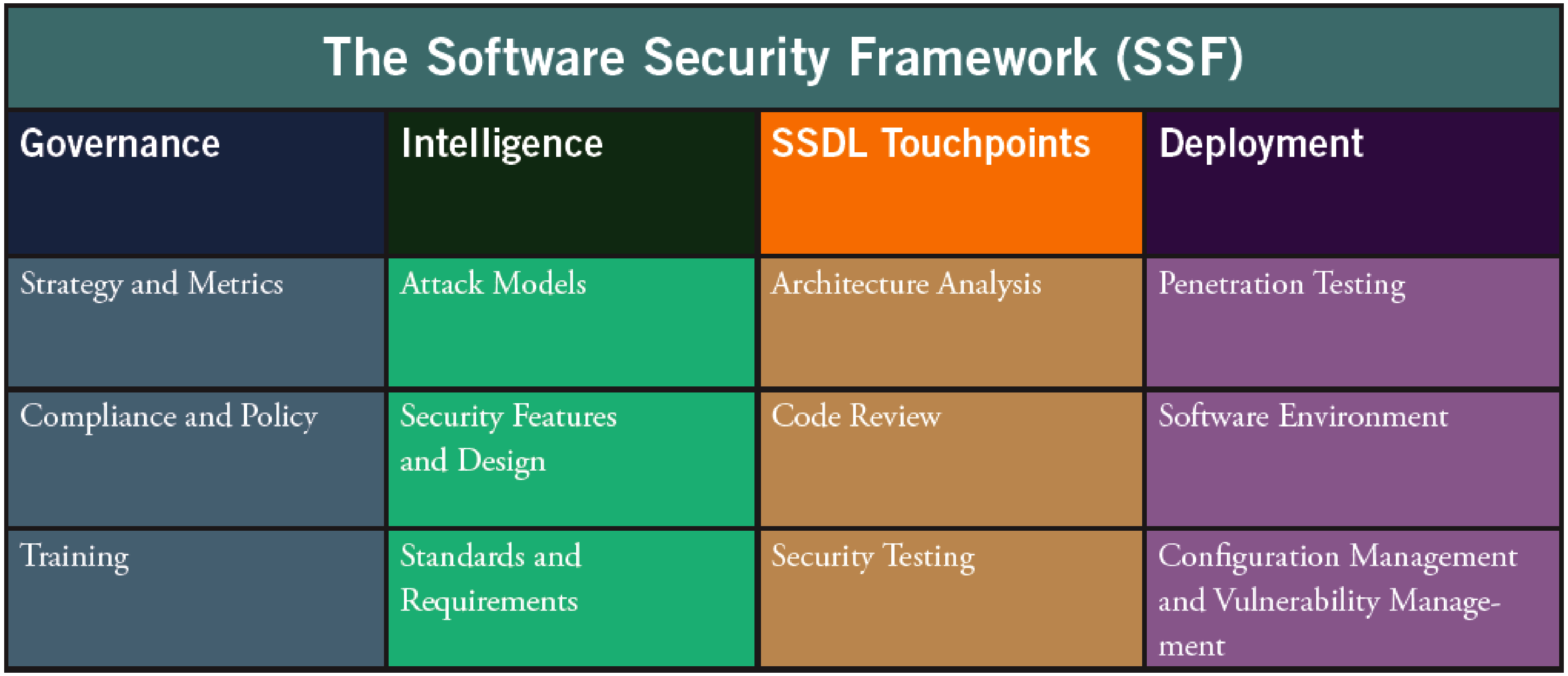
Developed by Gary McGraw and Sammy Migues, Cigital

BSIMM: Software Security Measurement

- As of 09/2011:
  - Real data from (42) real initiatives
  - 81 measurements
  - 11 over time
- McGraw, Chess, & Migues
- BSIMM4 coming soon

cigital PlexLogic FORTIFY CSO Minded security VIRTUALFORGE we harden your software

A 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 Manage-ment

- Four domains
- Twelve practices
- See informIT article on BSIMM website http://bsimm.com

Building BSIMM

- BSIMM1: Build a maturity model from actual data
  - Find some “volunteers”; started with 9
  - Conduct in-person executive interviews
  - Harmonize the data into unique activities
  - Provide objective and example for each activity
  - Populate the 12 practices to produce the model
  - Release under Creative Commons license for all to use
- BSIMM3: 42 firms as of September 2011 (http://bsimm.com)
  - 17 FI, 15 ISV, 10 high tech, 3 telecoms, 2 insurance, 2 energy, 2 media, 1 healthcare (counting overlap)
- BSIMM4: coming soon
  - 50+ firms, 13+ firms measured more than once

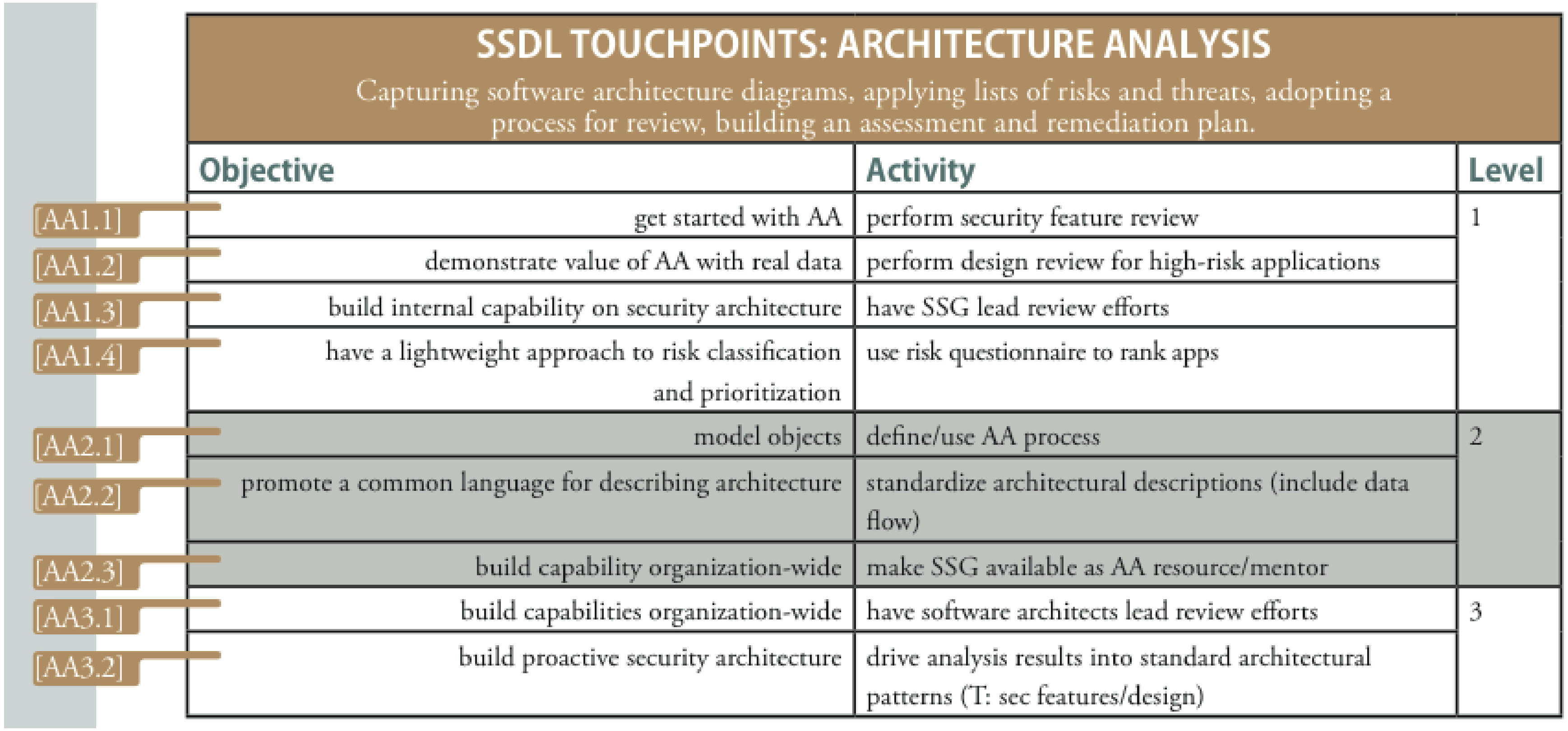
42 software security initiatives measured (09/2011)

- Adobe
- Aon
- Bank of America
- Capital One
- The Depository Trust & Clearing Corporation (DTCC)
- EMC
- Fannie Mae
- Fidelity
- Google
- Intel
- Intuit
- Mashery
- McKesson
- Microsoft

- Nokia
- QUALCOMM
- Sallie Mae
- SAP
- Scripps Networks Interactive
- Sony Mobile
- Standard Life
- SWIFT
- Symantec
- Telecom Italia
- Thomson Reuters
- Visa
- Vmware
- Wells Fargo
- Zynga

Plus 13 others

Architecture Analysis practice skeleton



SSDL TOUCHPOINTS: ARCHITECTURE ANALYSIS

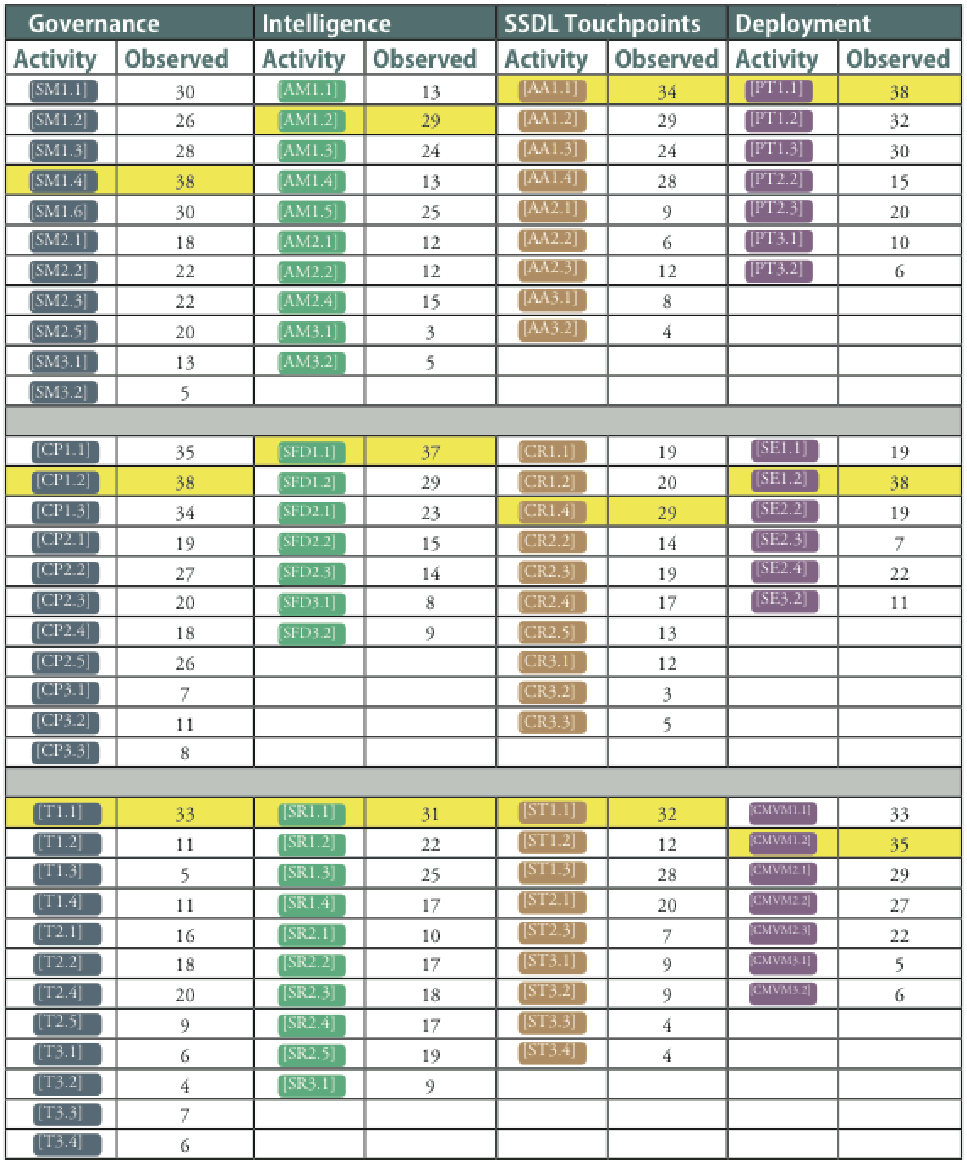
Capturing software architecture diagrams, applying lists of risks and threats, adopting a process for review, building an assessment and remediation plan.

	Objective	Activity	Level
[AA1.1]	get started with AA	perform security feature review	1
[AA1.2]	demonstrate value of AA with real data	perform design review for high-risk applications	
[AA1.3]	build internal capability on security architecture	have SSG lead review efforts	
[AA1.4]	have a lightweight approach to risk classification and prioritization	use risk questionnaire to rank apps	
[AA2.1]	model objects	define/use AA process	2
[AA2.2]	promote a common language for describing architecture	standardize architectural descriptions (include data flow)	
[AA2.3]	build capability organization-wide	make SSG available as AA resource/mentor	
[AA3.1]	build capabilities organization-wide	have software architects lead review efforts	3
[AA3.2]	build proactive security architecture	drive analysis results into standard architectural patterns (T: sec features/design)	

Example activity

[AA1.2] **Perform design review for high-risk applications.** The organization learns about the benefits of architecture analysis by seeing real results for a few high-risk, high-profile applications. If the SSG is not yet equipped to perform an in-depth architecture analysis, it uses consultants to do this work. Ad hoc review paradigms that rely heavily on expertise may be used here, though in the long run they do not scale.

BSIMM3 Scorecard

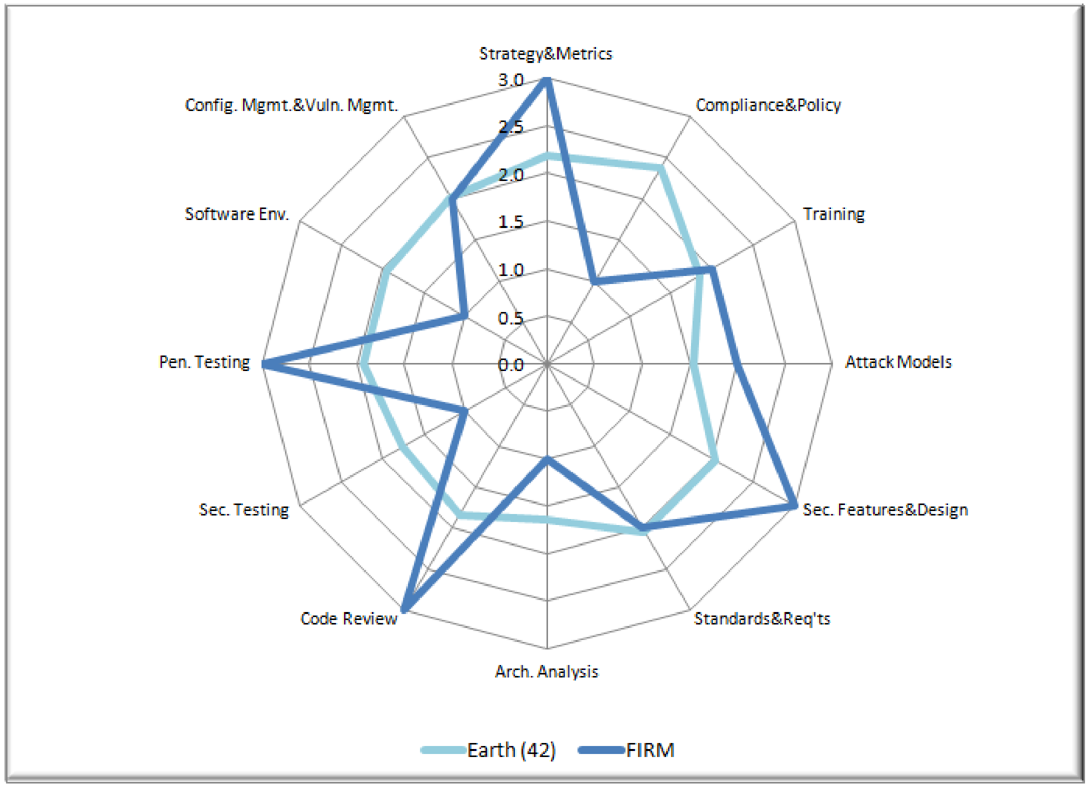


Governance		Intelligence		SSDL Touchpoints		Deployment	
Activity	Observed	Activity	Observed	Activity	Observed	Activity	Observed
[SM1.1]	30	[AM1.1]	13	[AA1.1]	34	[PT1.1]	38
[SM1.2]	26	[AM1.2]	29	[AA1.2]	29	[PT1.2]	32
[SM1.3]	28	[AM1.3]	24	[AA1.3]	24	[PT1.3]	30
[SM1.4]	38	[AM1.4]	13	[AA1.4]	28	[PT2.2]	15
[SM1.6]	30	[AM1.5]	25	[AA2.1]	9	[PT2.3]	20
[SM2.1]	18	[AM2.1]	12	[AA2.2]	6	[PT3.1]	10
[SM2.2]	22	[AM2.2]	12	[AA2.3]	12	[PT3.2]	6
[SM2.3]	22	[AM2.4]	15	[AA3.1]	8		
[SM2.5]	20	[AM3.1]	3	[AA3.2]	4		
[SM3.1]	13	[AM3.2]	5				
[SM3.2]	5						
[CP1.1]	35	[SFD1.1]	37	[CR1.1]	19	[SE1.1]	19
[CP1.2]	38	[SFD1.2]	29	[CR1.2]	20	[SE1.2]	38
[CP1.3]	34	[SFD2.1]	23	[CR1.4]	29	[SE2.2]	19
[CP2.1]	19	[SFD2.2]	15	[CR2.2]	14	[SE2.3]	7
[CP2.2]	27	[SFD2.3]	14	[CR2.3]	19	[SE2.4]	22
[CP2.3]	20	[SFD3.1]	8	[CR2.4]	17	[SE3.2]	11
[CP2.4]	18	[SFD3.2]	9	[CR2.5]	13		
[CP2.5]	26			[CR3.1]	12		
[CP3.1]	7			[CR3.2]	3		
[CP3.2]	11			[CR3.3]	5		
[CP3.3]	8						
[T1.1]	33	[SR1.1]	31	[ST1.1]	32	[CMVM1.1]	33
[T1.2]	11	[SR1.2]	22	[ST1.2]	12	[CMVM1.2]	35
[T1.3]	5	[SR1.3]	25	[ST1.3]	28	[CMVM2.1]	29
[T1.4]	11	[SR1.4]	17	[ST2.1]	20	[CMVM2.2]	27
[T2.1]	16	[SR2.1]	10	[ST2.3]	7	[CMVM2.3]	22
[T2.2]	18	[SR2.2]	17	[ST3.1]	9	[CMVM3.1]	5
[T2.4]	20	[SR2.3]	18	[ST3.2]	9	[CMVM3.2]	6
[T2.5]	9	[SR2.4]	17	[ST3.3]	4		
[T3.1]	6	[SR2.5]	19	[ST3.4]	4		
[T3.2]	4	[SR3.1]	9				
[T3.3]	7						
[T3.4]	6						

- 109 Activities
- 3 levels
- Top 12 activities
  - 69% cutoff
  - 29 of 42 firms
- Comparing scorecards between releases is interesting

How to use BSIMM

- A measuring stick for software security initiatives
- See what your peers are doing
- Compare firms, business units
- Study firm/BU change over time
- A lens on the state of software security
- Meet your peers at BSIMM events



BSIMM3 to BSIMM4

- BSIMM3 released September 2011 under creative commons
  - http://bsimm.com
  - Italian and German translations available
- BSIMM is a yardstick
  - Use it to see where you stand
  - Use it to figure out what your peers do
- BSIMM3→BSIMM4
  - BSIMM is growing
  - Target 50 firms
  - Target 100 measurements

BSIMM: Some Useful Resources

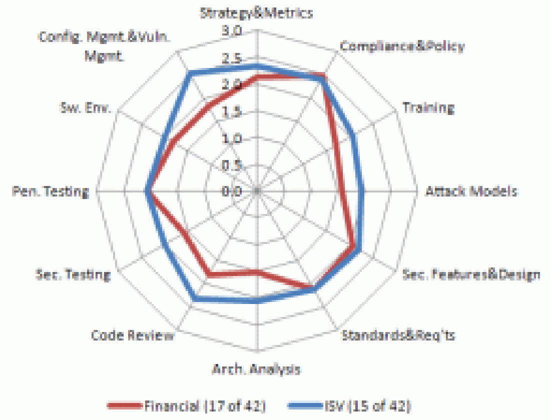
- http://bsimm.com/download/ (no registration required).
- Software [In]security: The Building Security In Maturity Model (BSIMM): http://www.informit.com/articles/article.aspx?p=1332285
- Software [In]security: BSIMM3: http://www.informit.com/articles/article.aspx?p=1755416
- Software [In]security: You Really Need a Software Security Group: http://www.informit.com/articles/article.aspx?p=1434903
- Software [In]security: Third-Party Software and Security: http://www.informit.com/articles/article.aspx?p=1809143
- Software [In]security: vBSIMM Take Two (BSIMM for Vendors Revised): http://www.informit.com/articles/article.aspx?p=1832574
- Software [In]security: Software Security Zombies: http://www.informit.com/articles/article.aspx?p=1739924

An Assurance Ecosystem

Developed by Dan Reddy EMC-2

One view as to how the pieces fit

BSIMM



Shows data congruence of security activities found in companies that were analyzed

THE Open GROUP
Making standards work®

- Standard that outlines best practices of ICT Providers to mitigate vs *tainted* & *counterfeit* products.
- Method to accredit Trusted Technology Providers.

SAFECode
Software Assurance Forum for Excellence in Code
Driving Security and Integrity

- Building secure products
- Prescriptive.
- How should I do it?
- Where should I start?

EMC-wide Standard with focus on Risk and Organization Maturity

Process Standard

- ✓ *Training*
- ✓ *Requirements*
- ✓ *Threat modeling*
- ✓ *Code scanning*
- ✓ *Security testing*
- ✓ *Documentation*
- ✓ *Assessment*
- ✓ *Vulnerability response*

PRODUCT SECURITY POLICY

Design Standard

- ✓ *Authentication & access control*
- ✓ *Logging*
- ✓ *Network security*
- ✓ *Cryptography and key management*
- ✓ *Serviceability*
- ✓ *Secure design principles*

Coding Standard

- ✓ *Input validation*
- ✓ *Injection protection*
- ✓ *Directory traversal protection*
- ✓ *Web and C/ C++ coding standards*
- ✓ *Handling secrets*

Source Code Standard

- ✓ *Sourcing software*
- ✓ *Source code protection*
- ✓ *Software delivery protection*
- ✓ *Product counterfeiting prevention*

ORG MATURITY LEVELS

- ➢ **Optimized**: *Risk is minimized*
- ➢ **Integrated**: *Risk is controlled*
- ➢ **Proactive**: *Risk is understood*
- ➢ **Reactive**: *Risk is unknown*

Gap assessment as part of standard product readiness process

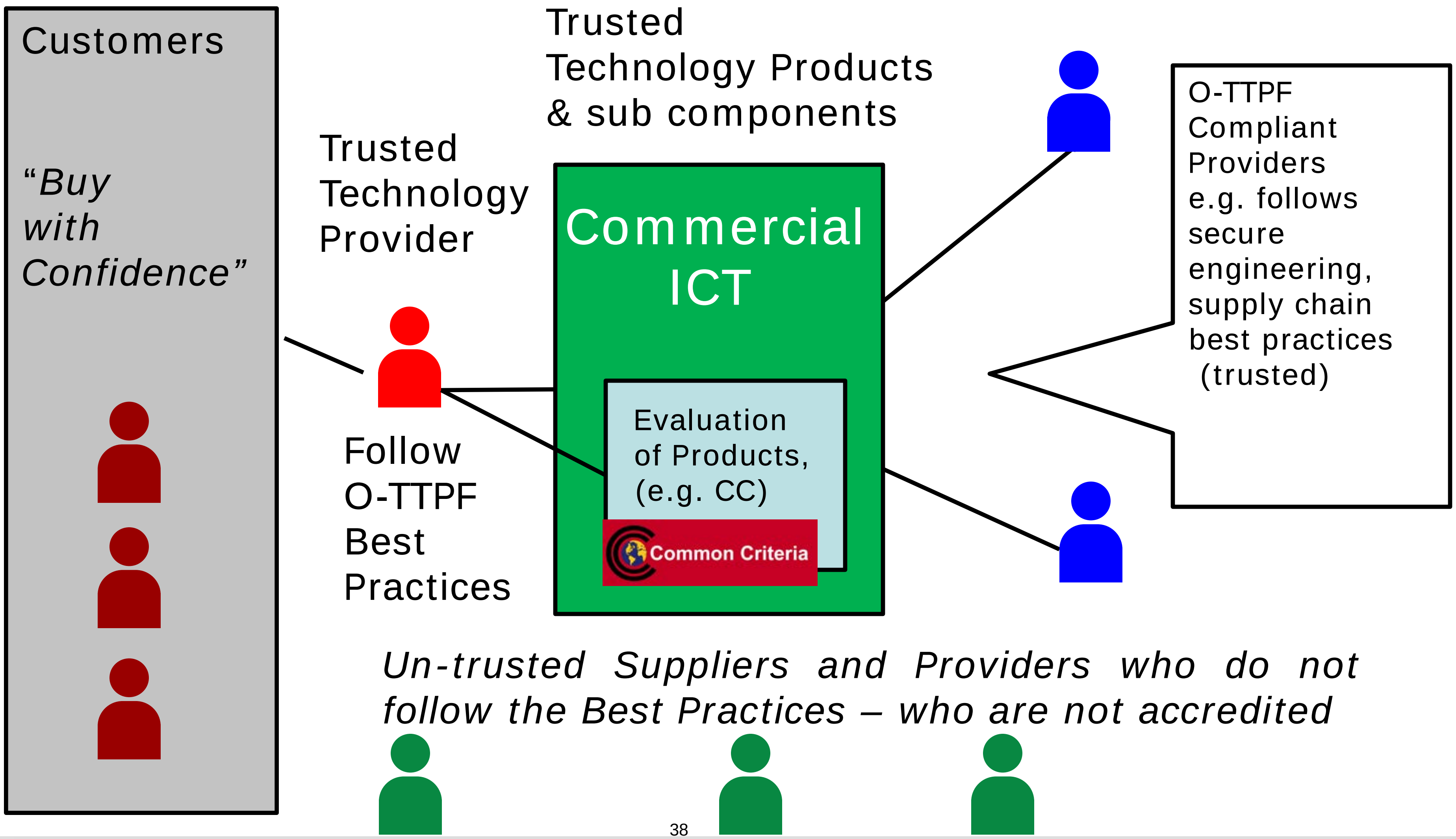
Security Development Lifecycle

PRODUCT RISK (4 levels)

- ➢ **Critical**: *Requires executive sign-off*
- ➢ **High**: *Requires remediation in next release*
- ➢ **Medium**: *Requires monitoring*
- ➢ **Low**

Customers Buy with More Confidence:

Providers & Suppliers Can Extend Supply Chain Integrity


Customers
"Buy with Confidence"
Trusted Technology Provider
Trusted Technology Products & sub components
Commercial ICT
Evaluation of Products, (e.g. CC)
Common Criteria
O-TTPF Compliant Providers e.g. follows secure engineering, supply chain best practices (trusted)
Follow O-TTPF Best Practices
Un-trusted Suppliers and Providers who do not follow the Best Practices – who are not accredited

Classifying Vulnerabilities: Some Useful Resources

- CVE: Common Vulnerabilities & Exposures Database
  - http://cve.mitre.org
- CWE: Common Weakness Enumeration
  - A community-developed dictionary of software weakness types
  - http://cwe.mitre.org/
- NVD: National Vulnerability Database
  - http://nvd.nist.gov
- Bugtraq mailing list: how to exploit & fix vulnerabilities
  - http://www.securityfocus.com/archive/1

Secure Coding: Some Useful Resources

- CERT Secure Coding Initiative
  http://www.cert.org/secure-coding/
- SANS Software Security Institute
  - http://www.sans-ssi.org/
- Open Web Application Security Project (OWASP)
  - http://www.owasp.org/
- Web Application Security Consortium (WASC)
  - http://www.webappsec.org/

Questions?

Looking Ahead: Lecture #3

I. Requirements Engineering

II. Security Requirements Engineering

III. Introduction to SQUARE

IV. SQUARE Demo Videos

Reading Assignment

- http://bsimm.com/download/ (no registration required)
- http://www.owasp.org/
- http://msdn.microsoft.com/en-us/library/ms995349.aspx
- https://buildsecurityin.us-cert.gov/daisy/bsi/resources/published/series/bsi-ieee/568.html
- https://buildsecurityin.us-cert.gov/bsi/articles/knowledge/sdlc.html

Case Study Team Formation

- Form teams of 4-5 people
- Each team should have 1 or more students working on a software development project that can be used as a software security case study
- The team members should have reasonably compatible schedules in order to accomplish the team work

Case Study Assignment #1

- (15%) Describe the project, and why it is a good software security project OR the changes that you have had to make to get it to be a good software security project.
- (20%) Describe the security lifecycle approach that you intend to use and the rationale for deciding on it. Why is it better than other approaches?
- (15%) What are the activities that this lifecycle approach supports?
- (20%) What is the underlying development model (e.g. Waterfall, Spiral, Agile)? Why is it a good model for this project?
- (15%) How well do the security activities fit with the selected development model?
- (15%) Compare your activities to the activities described in BSIMM3. Describe the similarities and differences. Are there important differences from a software security viewpoint?
- Turn this in on Blackboard BEFORE the next class.