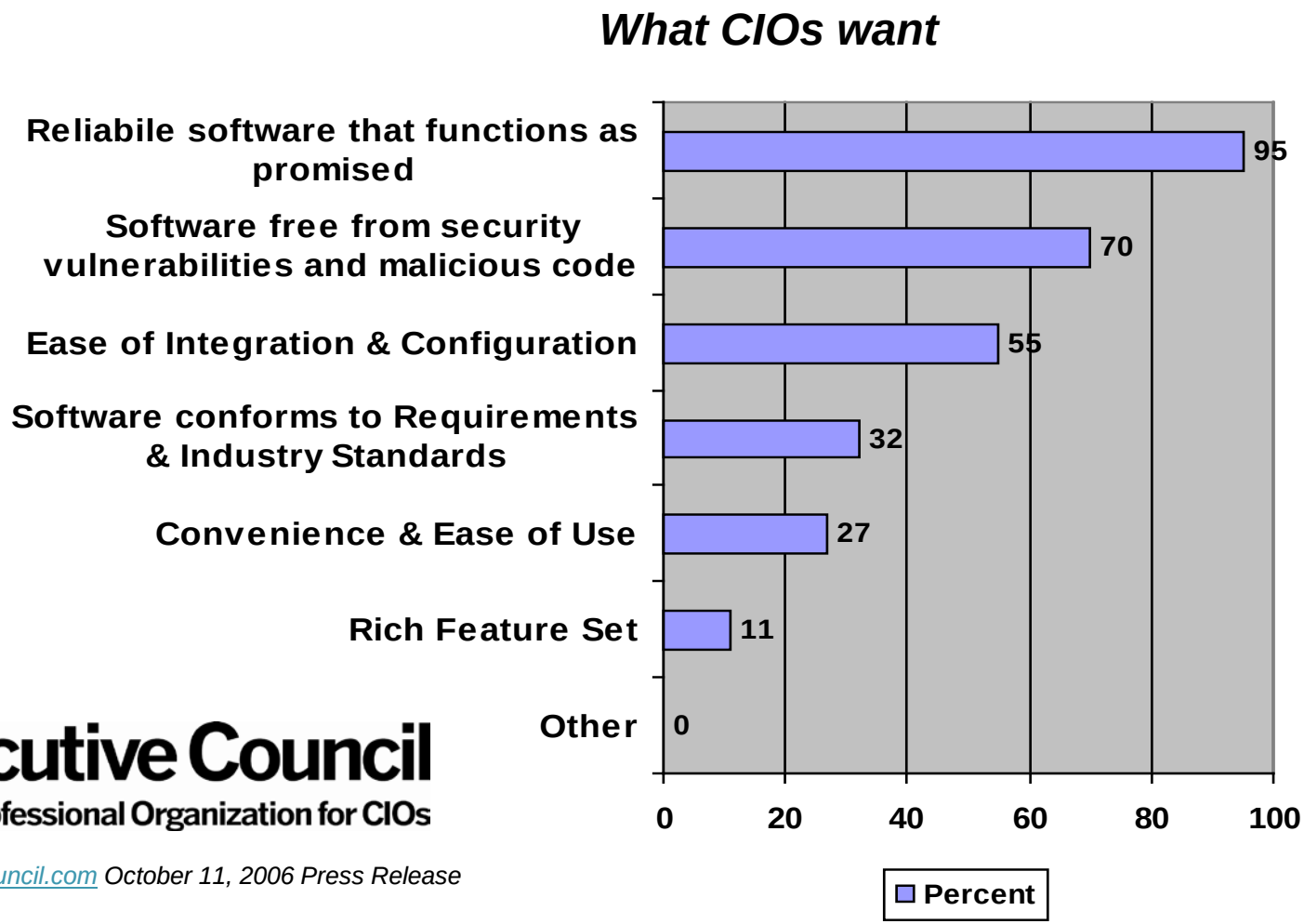
Why do developers make dangerous software errors?

Michele Moss
NDIA CMMI® Technology Conference
November 15, 2011

Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

Requirements for secure code are implicitly and not explicitly stated



CIO Executive Council
The Professional Organization for CIOs

https://www.cioexecutivecouncil.com *October 11, 2006 Press Release*

“Defacto” security requirements in NIST 800-53 do not explicitly require developers to produce secure code

- Technical
  - AC-2 Account Management
  - AC-3 Access Enforcement
  - AC-4 Information Flow Enforcement
- Management
  - RA-5 Vulnerability Scanning
- Operational
  - CM-7 Least Functionality
  - SI-3 Malicious Code Protection
  - SI-10 Information Input Validation

Technical controls in NIST 800-53 contribute to application security

- AC-7 Unsuccessful Login Attempts
- AC-10 Concurrent Session Control
- AC-11 Session Lock
- AC-14 Permitted Actions without Identification or Authentication
- AC - 16 Security Attributes
- AC-17 Remote Access
- AC-20 Use of External Information Systems
- AU-2 Auditable Events
- AU-3 Content of Audit Records
- AU-4 Audit Storage Capacity
- AU-9 Protection of Audit Information
- IA-3 Device Identification and Authentication
- IA-4 Identifier Management
- IA-5 Authenticator Management
- IA-6 Authenticator Feedback
- IA-7 Cryptographic Module Authentication
- IA - 8 Identification and Authentication (Non-Organizational Users)
- SC-2 Application Partitioning
- SC-3 Security Function Isolation
- SC-4 Information in Shared Resources
- SC-7 Boundary Protection
- SC-8 Transmission Integrity
- SC-10 Network Disconnect
- SC-11 Trusted Path
- SC-12 Cryptographic Key Establishment and Management
- SC-24 Fail in Known State
- SC-25 Thin Nodes
- AC-18 Wireless Access
- AC-19 Access Control for Mobile Devices
- SC-9 Transmission Confidentiality
- SC-13 Use of Cryptography
- SC-28 Protection of Information at Rest
- SC-23 Session Authenticity
- AC-5 Separation of Duties
- AC-6 Least Privilege

Operational controls in NIST 800-53 contribute to application security

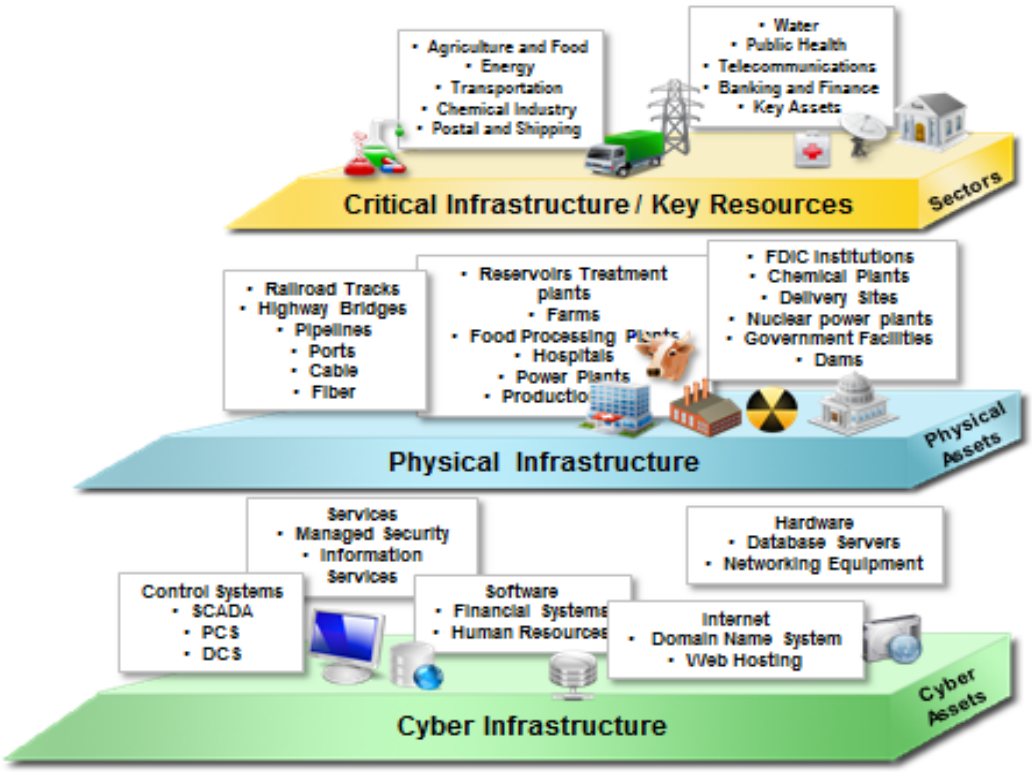
- SI-1 System and Information Integrity Policy and Procedures
- SI-2 Flaw Remediation
- AT-2 Security Awareness
- AT-3 Security Training
- CM-3 Configuration Change Control
- CM-4 Security Impact Analysis
- CM-5 Access Restrictions for Change
- SI-4 Information System Monitoring
- SI-6 Security Functionality Verification
- SI-7 Software and Information Integrity
- CA-5 Plan of Action and Milestones
- RA-3 Risk Assessment
- SA-2 Allocation of Resources
- SA-3 Life Cycle Support
- SA-4 Acquisitions
- SA-6 Software Usage Restrictions
- SA-10 Developer Configuration Management
- SA-11 Developer Security Testing
- SA-12 Supply Chain Protection
- SA-13 Trustworthiness
- PM-1 Information Security Program Plan
- PM-3 Information Security Resources
- PM-6 Information Security Measures of Performance
- PM-7 Enterprise Architecture
- PM-9 Risk Management Strategy
- SA-8 Security Engineering Principles
- CM-6 Configuration Settings
- CM- 8 Information System Component Inventory
- SI-11 Error Handling
- SI-12 Information Output Handling and Retention

Key questions

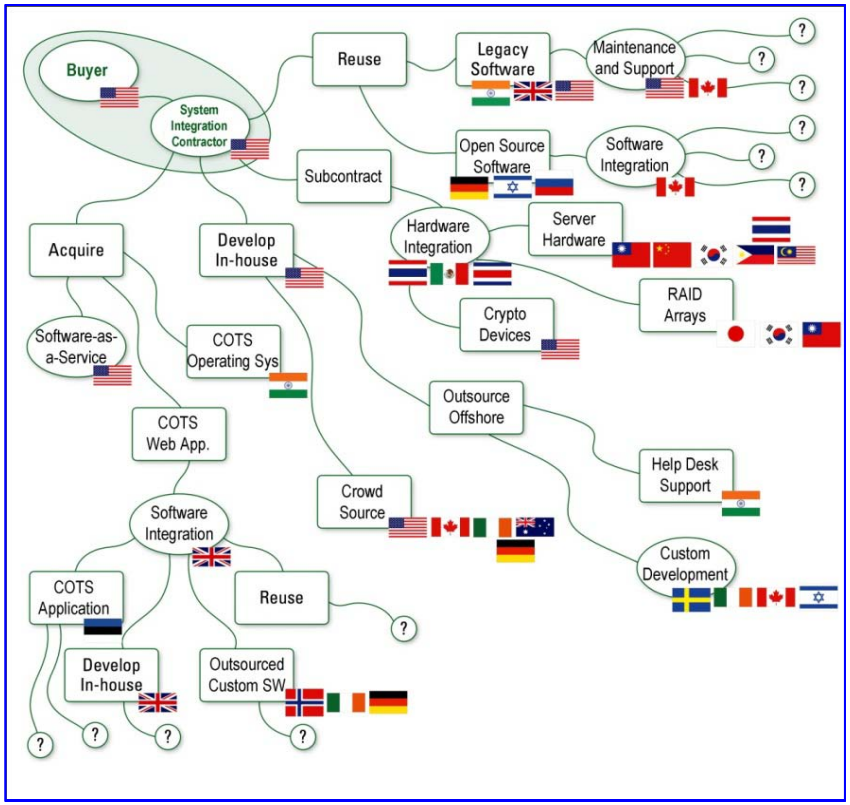
- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

Perspective on technology today

Technology is an integral part of our lives



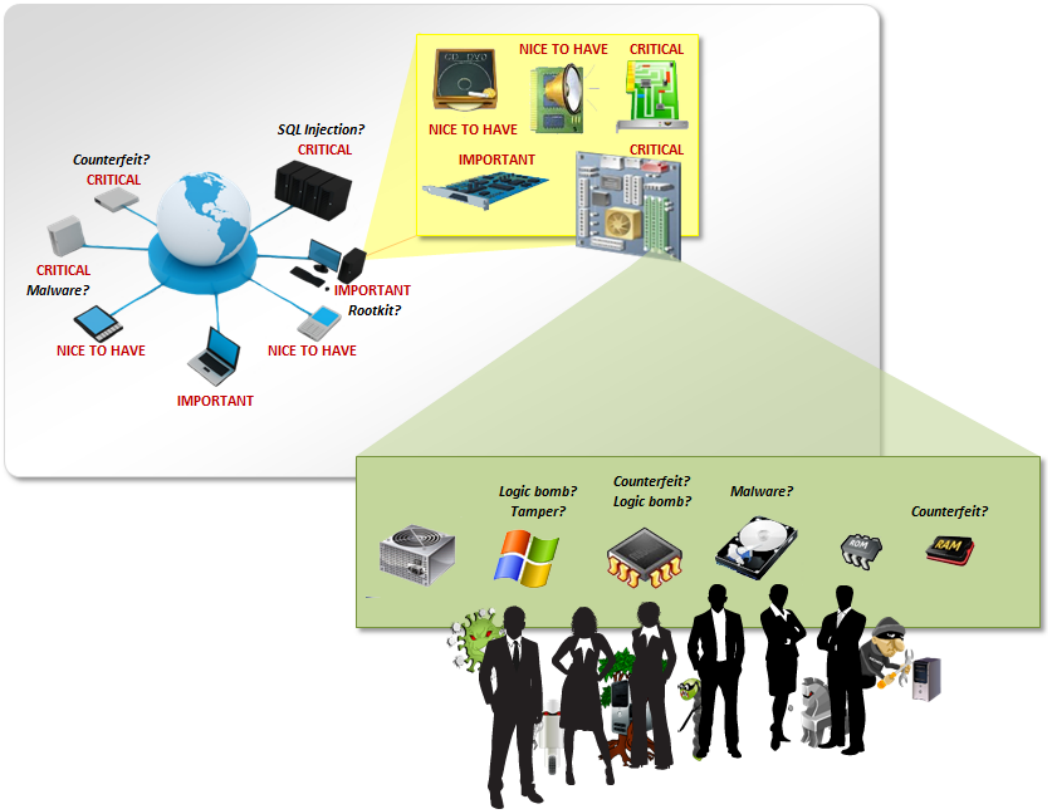
IT and Communications products are assembled, built, and transported by multiple vendors around the world



Malicious actors are taking advantage of abundant opportunities to tamper with and sabotage products …

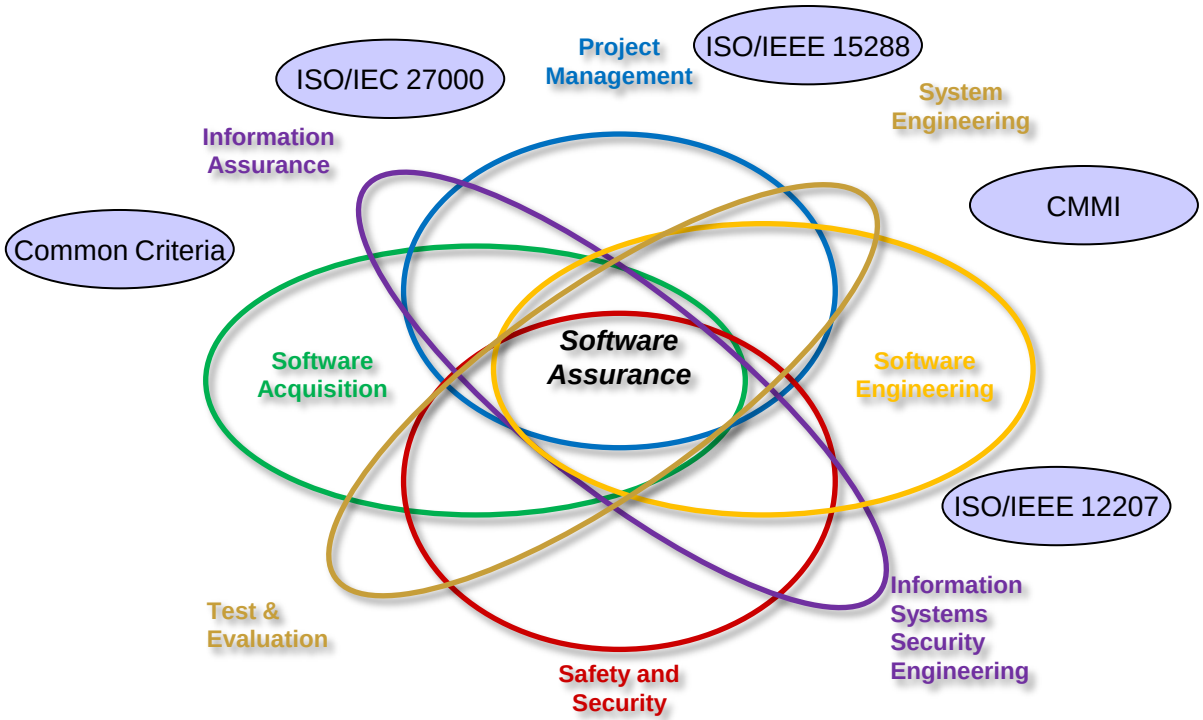
What commonalities exist?	**83%** of victims were targets of opportunity **92% of attacks were not highly difficult** **86%** were discovered by a third party **96% of breaches were avoidable through simple or intermediate controls**
How do breaches occur?	**50% utilized some form of hacking** 49% incorporated malware (lower percentages included physical attacks, privilege misuse, and social tactics)

**** Source – 2011 Verizon Data Breach Investigations Report***



… ultimately compromising system integrity and operations

SwA requires multi-disciplinary collaboration



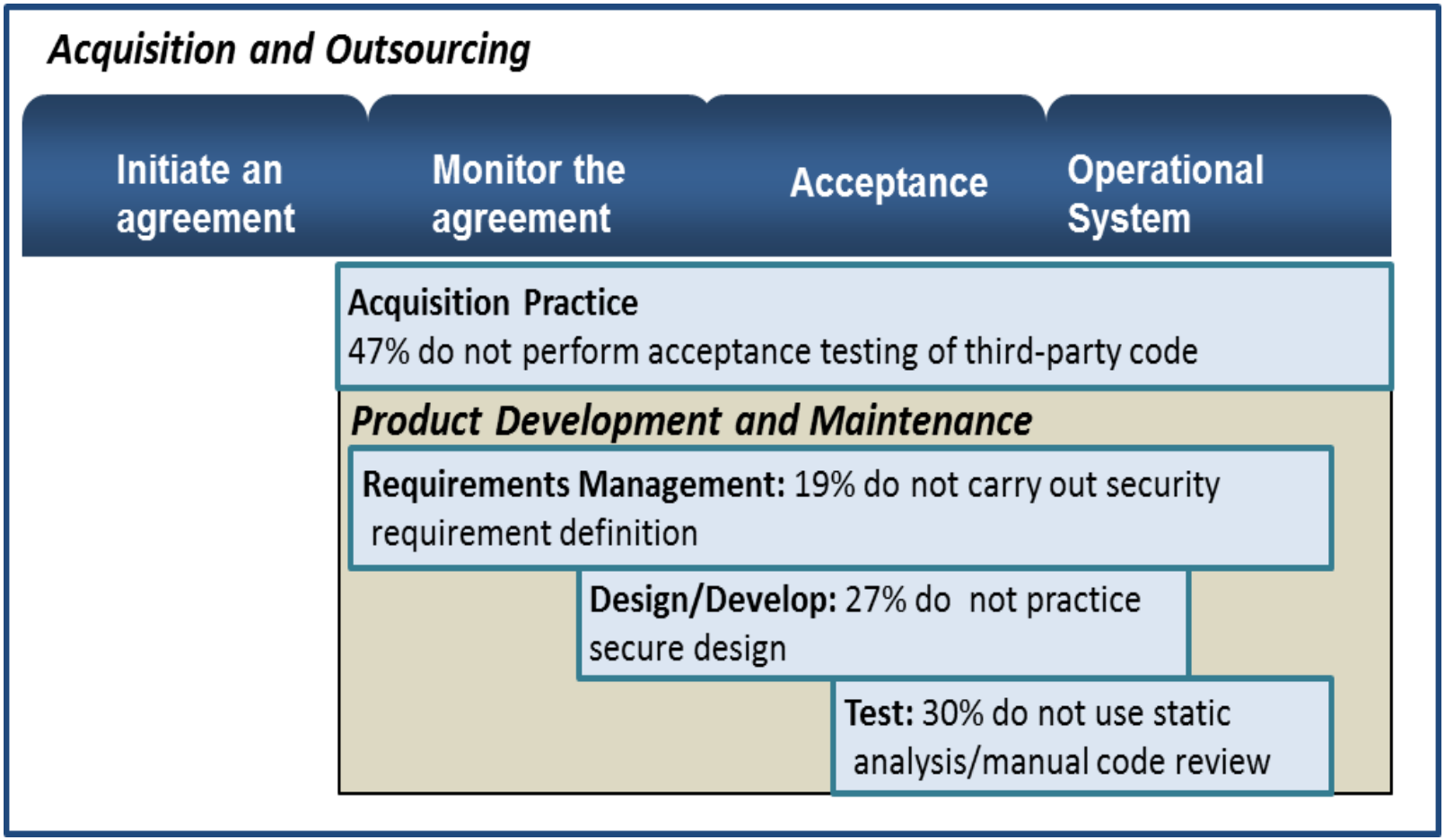
Communication Challenges

- Vocabulary
- Reserved Words
- Priorities
- Perspective
- Experience
- Objectives
- Drivers
- Risks

Source: https://buildsecurityin.us-cert.gov/swa/procresrc.html

Without a common language we cannot communicate across disciplines

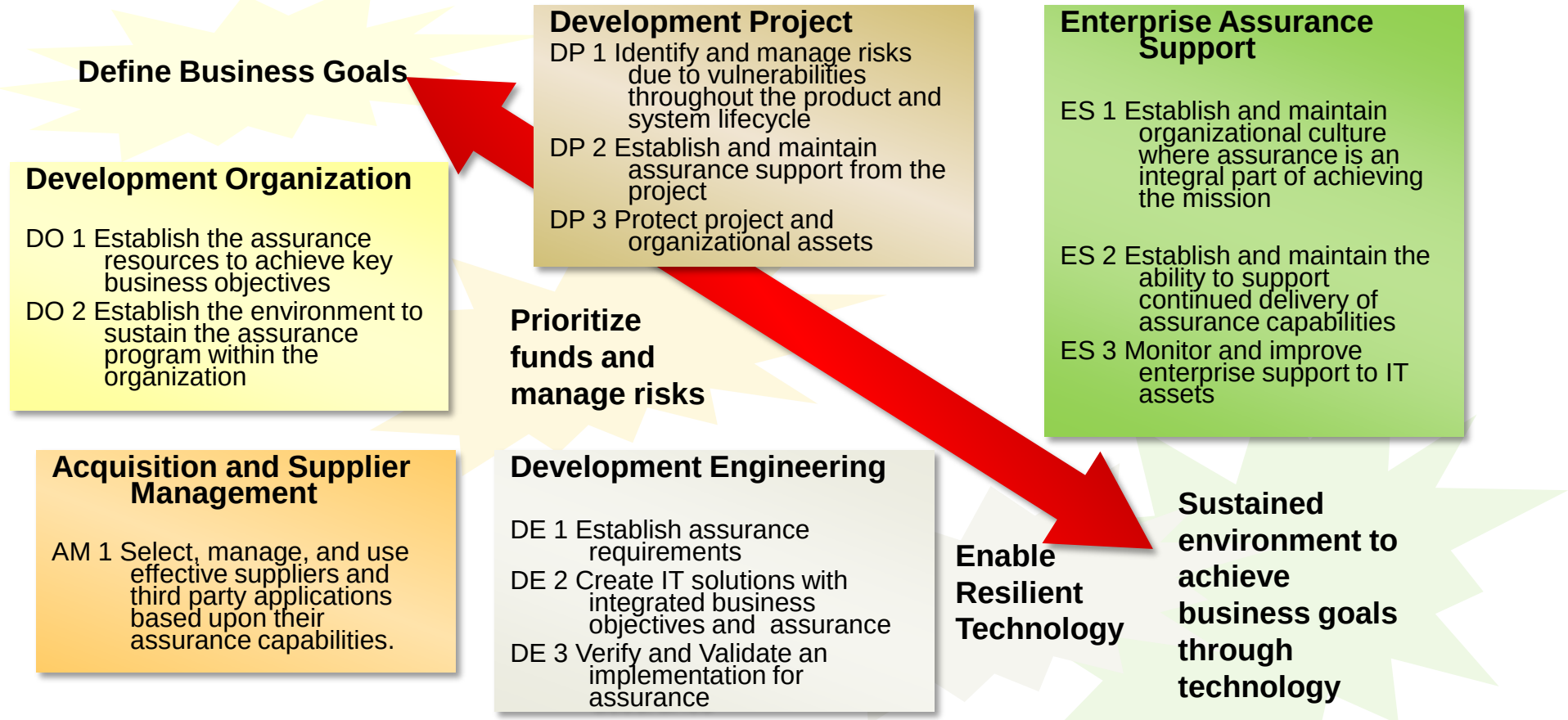
Acquirers of IT products and services trust that suppliers are addressing cyber security without validating



Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

Communication across organizational stakeholders is critical to addressing SwA challenges



The Assurance PRM Is A Holistic Framework that connects CMMI and RMM to facilitate communication

https://buildsecurityin.us-cert.gov/swa/proself_assm.html

A majority of SwA best practices focus on developer-centric audiences from a security point of view

Development Organization

DO 1 Establish the assurance resources to achieve key business objectives

DO 2 Establish the environment to sustain the assurance program within the organization

Development Project

DP 1 Identify and manage risks due to vulnerabilities throughout the product and system lifecycle

DP 2 Establish and maintain assurance support from the project

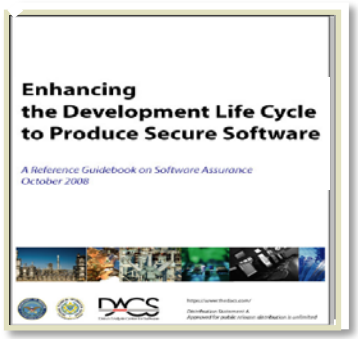
DP 3 Protect project and organizational assets

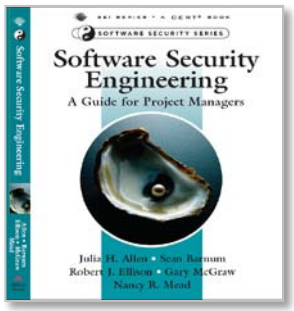
Development Engineering

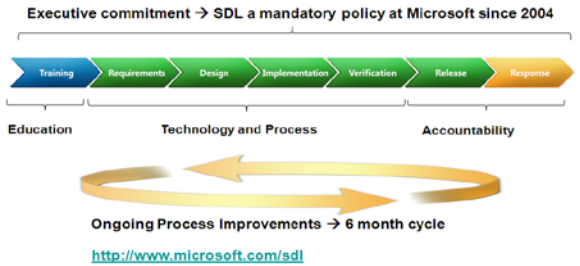
DE 1 Establish assurance requirements

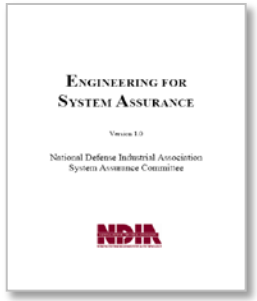
DE 2 Create IT solutions with integrated business objectives and assurance

DE 3 Verify and Validate an implementation for assurance

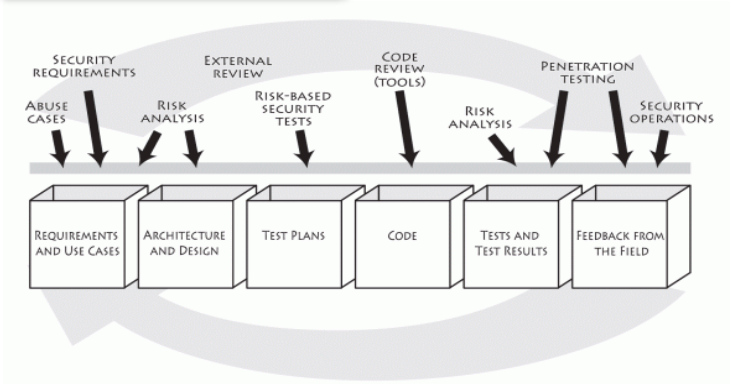








Assurance for CMMI ®

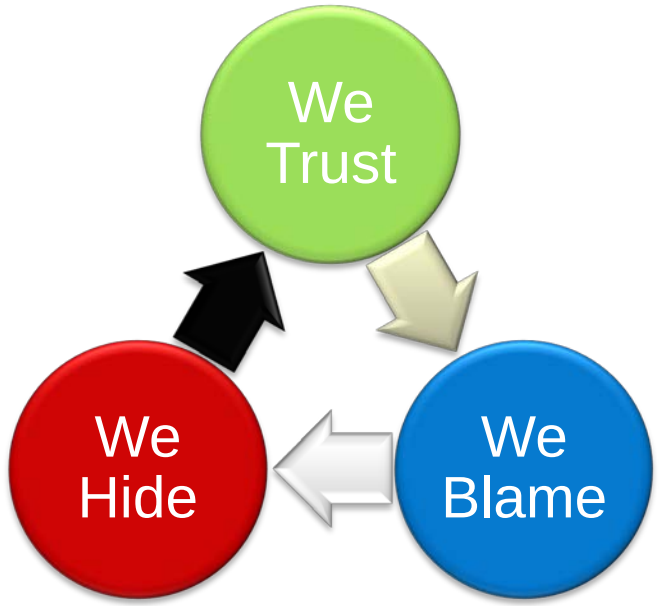


Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

100 apps written by 100 developers at 100 companies

- 83 apps have serious vulnerabilities
- 72 apps have cross site scripting
- 40 apps have SQL Injection
- 100 apps contain code of unknown origin
- 90 apps use un-patched libraries with known flaws
- 5 apps have had a scan or pentest
- 1 app has had a manual security code review
- 0 apps provide any visibility into security



Why

- **1 company has a responsible appsec program**
- **1 developer has any security training**

Adapted from: The Open Web Application Security Project ,Jeff Williams, Aspect Security, SWA Forum Sept 2010

Implementation lessons learned from some of the 1/100 companies that implement SwA successfully

Who
- Secure development SMEs
- Developers

What
- Measure progress (training, secure code reviews, security change requests)
- Internal policy

When
- During product development process
- During Leadership discussions
- As part of development and acquisition reviews

Where
- IT Development Organizations
- IT Acquisition Organizations
- IT Integrator Organizations

Courtesy of September 2010 SwA Panel SwA Practices – Getting to Effectiveness in Implementation

Why
- Customer pressure
- Reaction to an incident

Why Not
- Compliance drivers don't exist
- Focus is on systems and networks
- Secure software training is not given to developers and architects

How
- ***Executive leadership commitment***
- Translate ROI to project manager vocabulary (cost, schedule, quality)
- Start small and build
- Use coding standards
- Empower secure development to prevent a product from moving to the next milestone
- Avoid creating a new language
- Leverage what is already known
- Increase automation of menial tasks

Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

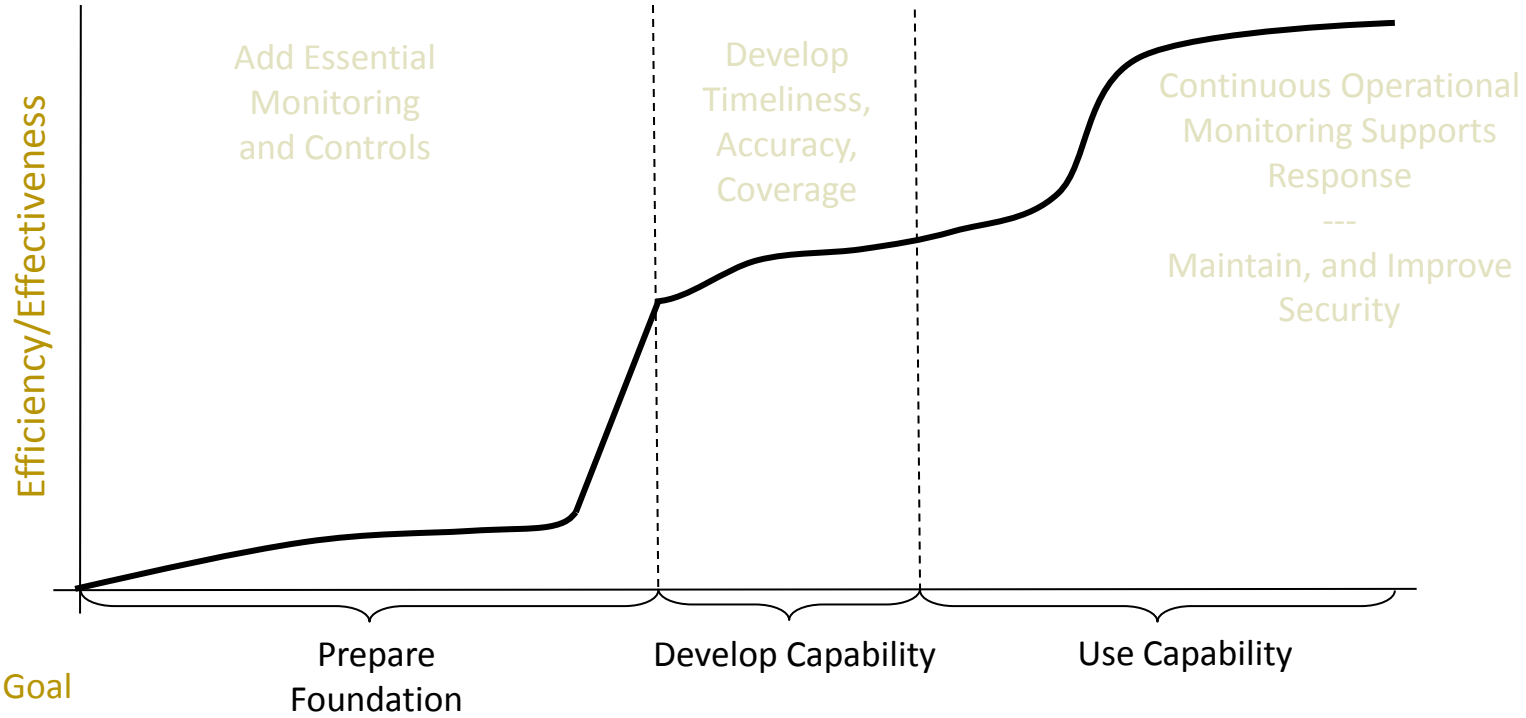
Measure, measure, and measure again

"*The only man I know who behaves sensibly is my tailor; he takes my measurements anew each time he sees me. The rest go on with their old measurements and expect me to fit them*."

- George Bernard Shaw

Source: www.CartoonStock.com

Robust measurement does not happen overnight and requires foundational capabilities in place to be effective



LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Courtesy of Matt Coose, DHS

Critical success factor – long-term management commitment, focus, and appropriate expectations

Problem

- Senior management has not expressed explicit support for the program
- There is no long term funding and resource commitment
- There is a perception in the organization that measurement and monitoring are temporary and “will be done” at some point
- Security improvements are expected within a very short time period and are expected to be easily directly related to measurement and monitoring
- Program is expected to be perfectly planned and executed as if the organization has done this a million times and has the process down perfectly
- Turnover and changes in roles breaks up continuity
- Accountability for metrics is difficult to assign

Solution

- Obtain senior management commitment before you start
- Work across the organization with the stakeholders to make them a part of the solution
- Iterate the program to measure critical things
- Structure the program to begin with quick wins to continuously demonstrate increase in value
- Manage expectations continuously – explain that the long term focus is critical
- Assign roles, train your responsible parties, and communicate that continuity is key for success
- Emphasize positive reinforcement – if everyone is failing, nobody will cooperate and the program will fail

Critical success factor – realistic and well thought out data collection strategy

Problem

- Data sources not well known, well defined, and therefore not considered
- Desired data is not obtainable
- Automated data sources are dispersed throughout the organization and data is difficult to consolidate
- Authoritative data sources do not exist
- Data is collected inefficiently or incorrectly
- Collection deemed too difficult too early
- Difficulty or inability to capture historical data

Solution

- Identify all available automated and manual data sources
- Define data that you need and compare with what available
- Create data collection strategy that would balance the need for data with the current state and plan for deliberately expanding data sources and measures
- Define future changes to processes/tools or requirement for new tools early in the process and refresh as you learn more
- Use feasibility of data collection as one of the criteria for metrics selection
- Train your data collectors and information owners about what you need and then ask for it repeatedly

Critical success factor – effective use of the measures to improve security

Problem

- Metrics analysis is not prioritized according to strategic goals, risks, ROI, or other explicit criteria agreed upon by the organization
- Wrong types of measures are distributed to wrong stakeholders
- Measures are collected for compliance and are not used to improve security
- Metrics data is not used for risk-based decision making

Solution

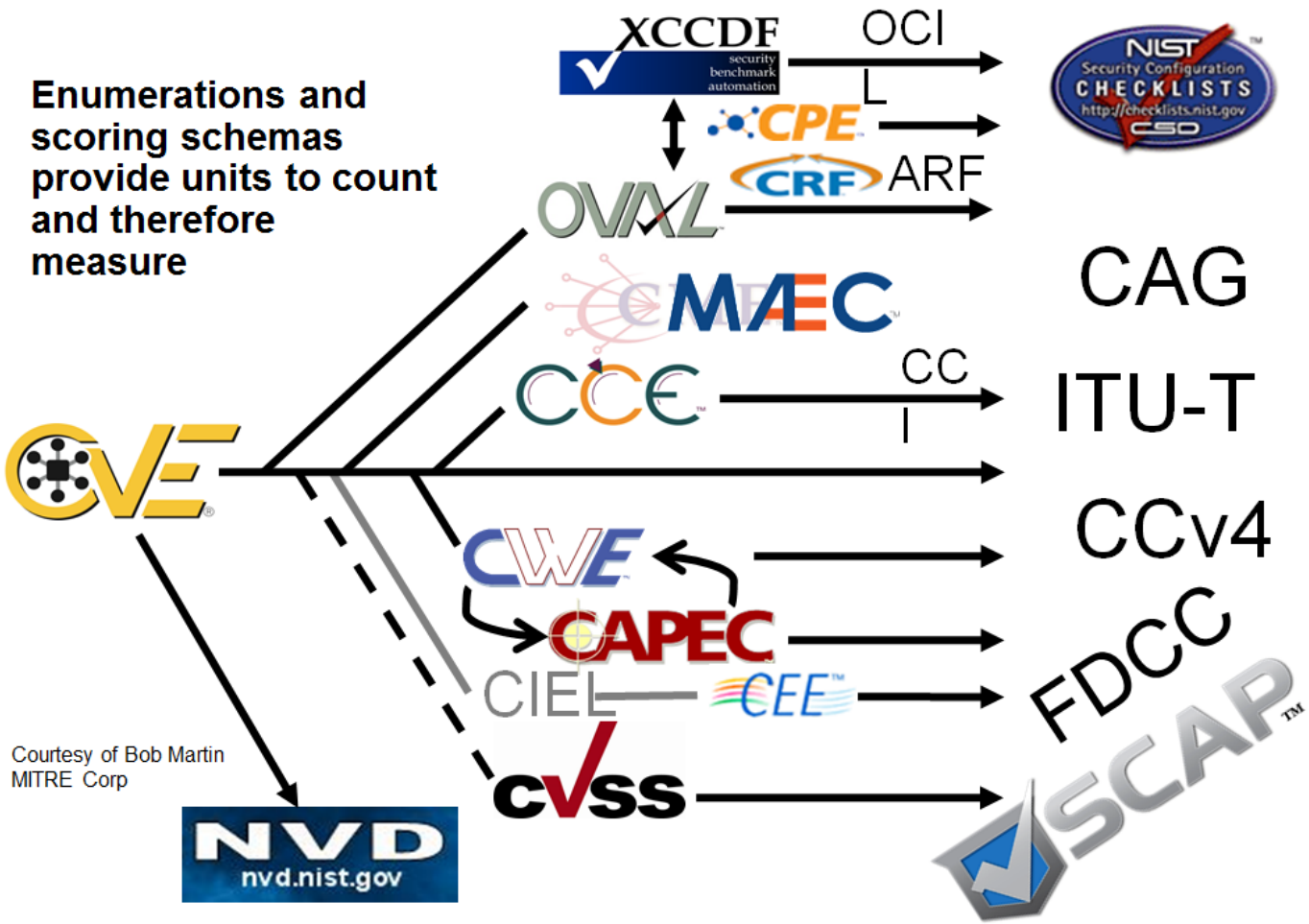
- Develop criteria for prioritizing and scoring measures early in the program and reconsider every time you expand the program
- Define who gets what data and reassess periodically – ask your customers if it is still useful
- If metrics are not used for decision making and improvement, drop them
- Communicate, communicate, communicate

Security control measures

- Percent of new systems that have completed certification and accreditation (C&A) prior to their implementation (NIST SP 800-53 Control: CA-6: Security Accreditation)
- Percent of employees who are authorized access to information systems only after they sign an acknowledgement that they have read and understood rules of behavior (NIST SP 800-53 Controls – PL-4: Rules of Behavior and AC-2: Account Management)
- Percent of the agency's information system budget devoted to information security (NIST SP 800-53 Controls – SA-2; Allocation of Resources)

Security Control Measures address compliance with the end state of the system, but not the underlying processes, structures, and code

Measurement for secure code requires understanding code level attributes …



Measurement for secure code involved understanding the effectiveness of implemented processes

- Acquisition
  - Number and percent of acquisition discussions that include SwA representative
  - Number and percent of contracting officers who received training in the security provisions of the FAR
  - Percent of documented Supplier claims verified through testing, inspection, or other methods
  - Number and percent of relevant high impact vulnerabilities (CVEs) present in the system
- Testing
  - Number and percent of tests that evaluate application response to misuse, abuse, or threats
  - Number and percent of tests that attempt to subvert execution or work around security controls
  - Percent of untested source code related to security controls and SwA requirements

How to apply…. success is ***simple***

Get Management Support

- Scope out measurement
- Obtain tangible support for security measures development and use at every management level
- Maintain support through regular reporting to stakeholders, tailored to their levels
  - Address their goals
  - Refine detail further up the management chain
  - Use Dashboards & Reports to endure

Start Small

- Expand your project/program cost, schedule, quality, and growth measures to cover security
- Start with a manageable, small set of security measures adding more measures as the project matures
- Leverage existing industry lists and select applicable measures
- Use a framework to translate measures from industry lists into your organization's approach
- Train data collectors to think in terms of metrics

Measure Behavior

- Identify and measure best and worst process and practice behaviors as well as results
- Create, display, and report measures to influence appropriate behavior
- Take advantage of unintended consequences produced by measurement
- Reuse measures where possible

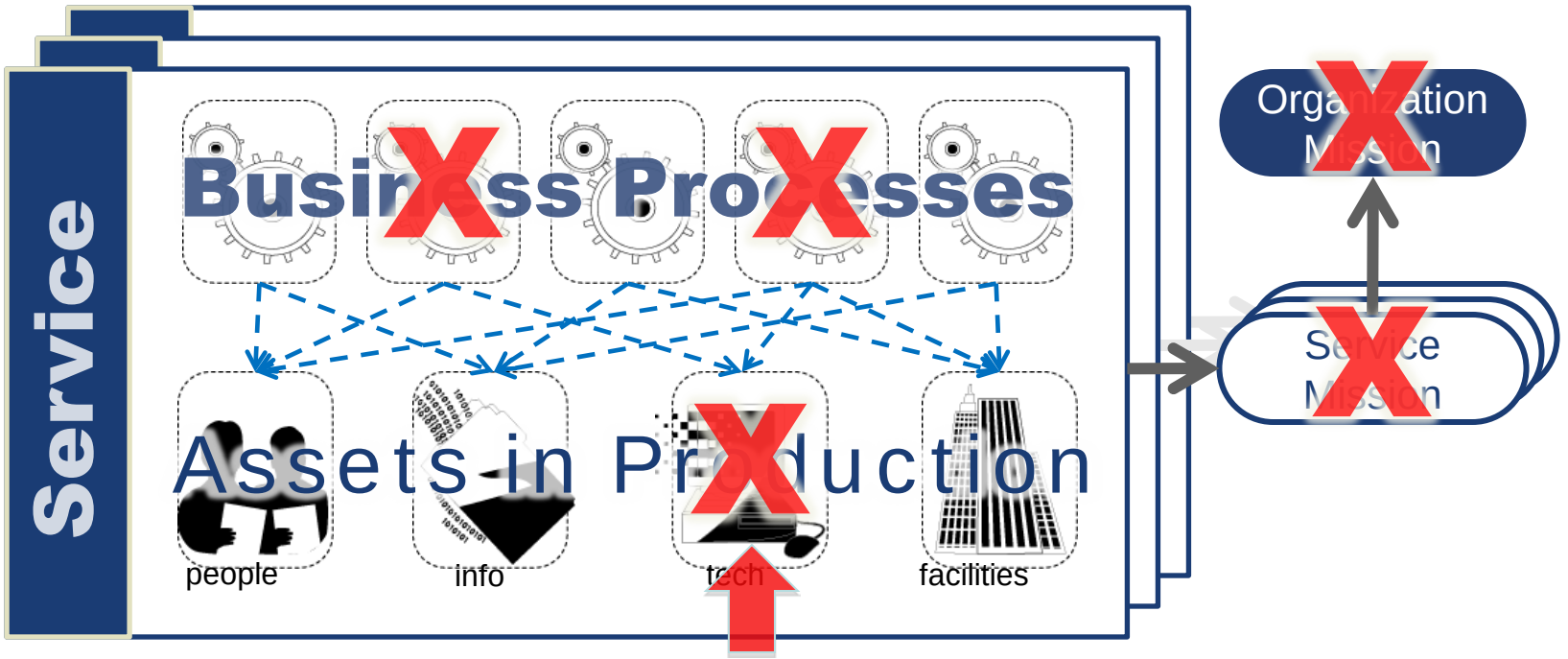
Incorporate security measures into your existing measurement activities

Source: http://www.psmsc.com/Downloads/TechnologyPapers/SwA%20Measurement%2010-08-08.pdf, accessed 4/10/09

Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next Steps?

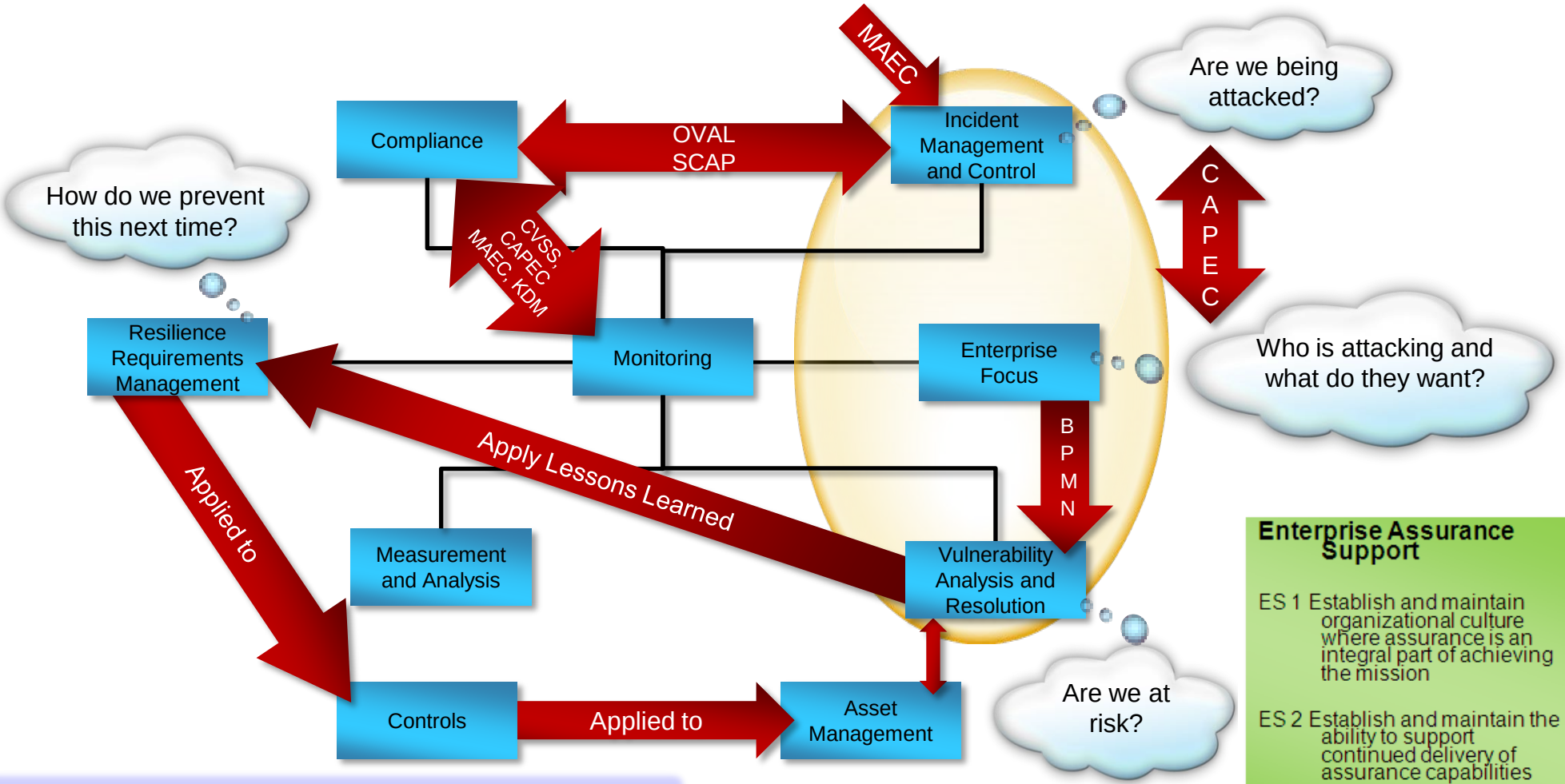
Business functions rely on accurate and reliable information from technology that functions as intended (and only as intended)

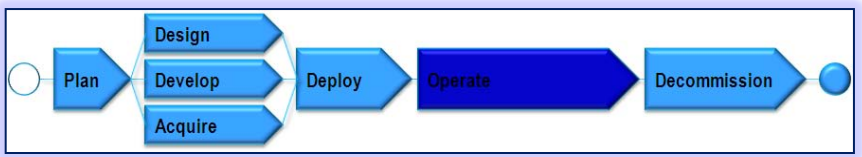


CERT | Software Engineering Institute | Carnegie Mellon.

Adapted from: November 2009 SwA Forum-Evolution in SwA Processes Panel – David White, SEI

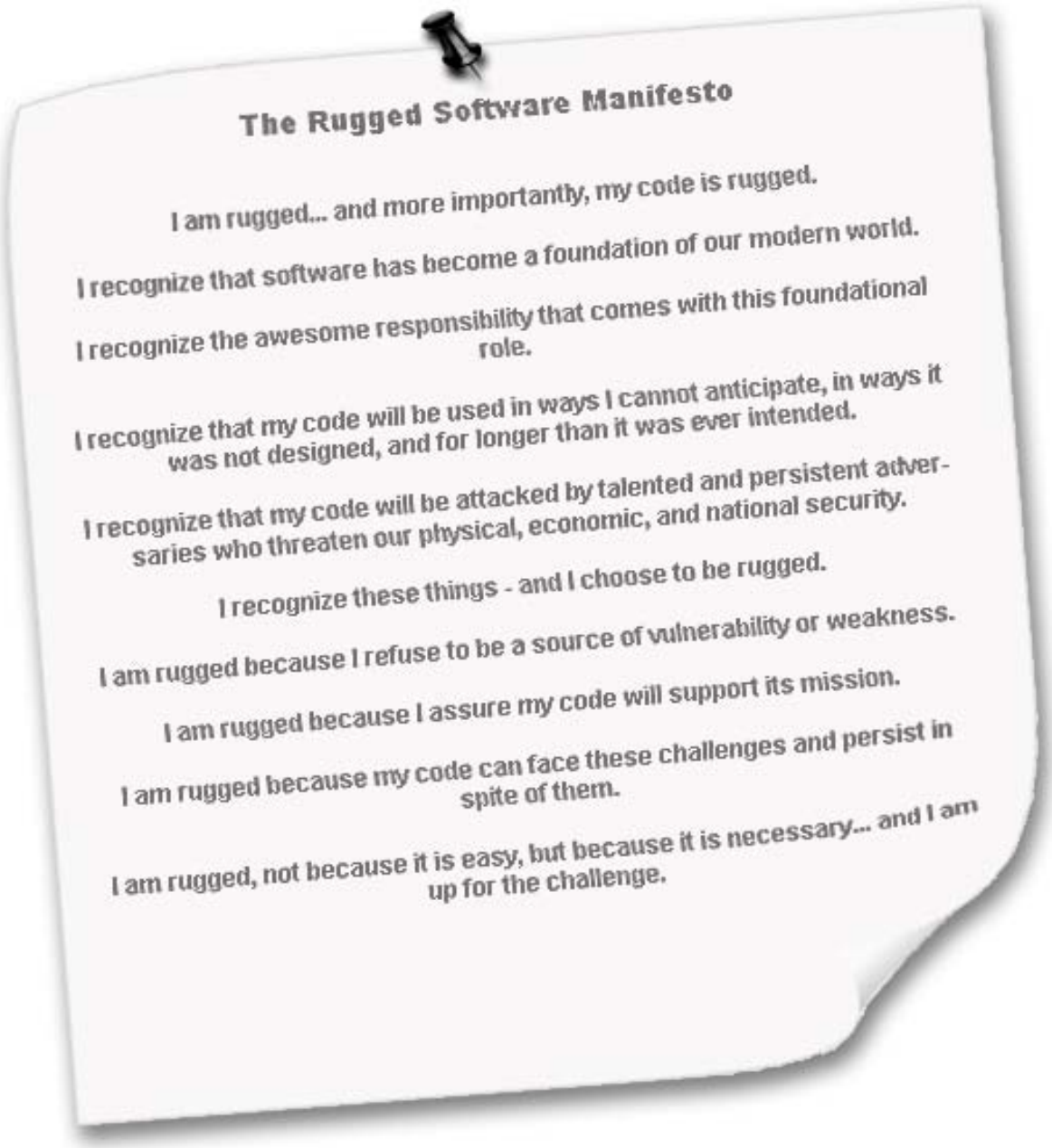
Potential impacts from threats to business functions can be understood by communicating software level vulnerabilities





Adapted from September 2010 SwA Forum, CERT RMM for Assurance , Lisa Young, SEI

http://www.ruggedsoftware.org/



The Rugged Software Manifesto

I am rugged... and more importantly, my code is rugged.

I recognize that software has become a foundation of our modern world.

I recognize the awesome responsibility that comes with this foundational role.

I recognize that my code will be used in ways I cannot anticipate, in ways it was not designed, and for longer than it was ever intended.

I recognize that my code will be attacked by talented and persistent adversaries who threaten our physical, economic, and national security.

I recognize these things - and I choose to be rugged.

I am rugged because I refuse to be a source of vulnerability or weakness.

I am rugged because I assure my code will support its mission.

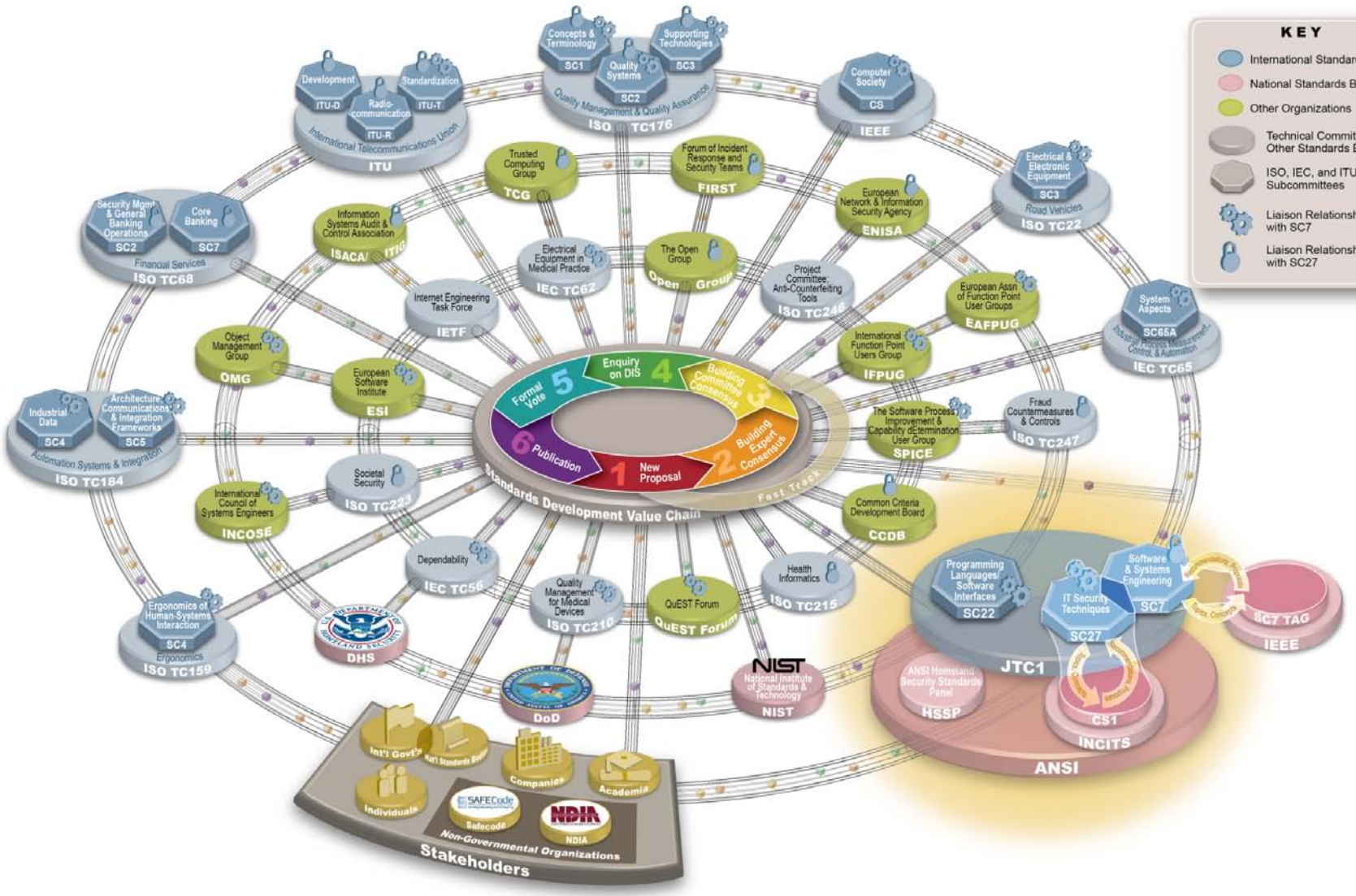
I am rugged because my code can face these challenges and persist in spite of them.

I am rugged, not because it is easy, but because it is necessary... and I am up for the challenge.

Key questions

- Who asks developers to develop secure code?
- Do developers know why they need to develop secure code?
- What should developers do/not do to develop secure code?
- Can developers develop secure code by themselves?
- How do developers know they have developed secure code?
- Why should developers care?
- Next steps?

Cyber security and software assurance standard development organization landscape



SC22 – Programming Languages, ISO/IEC TR 24772, Programming Language Vulnerabilities

- Targets building software that is inherently less vulnerable through improving the programming languages, or, at least, improve the usage of them in coding
- A catalog of 60+ issues that arise in coding when using any language and how those issues may lead to security and safety vulnerabilities
- Cross-referenced to CWE
- Each discussion includes
  - Description of the mechanism of failure
  - Recommendations for programmers: How to avoid or mitigate the problem.
  - Recommendations for standardizers: How to improve programming language specifications.

Courtesy of Jim Moore, MITRE

ISO/IEC 27036: Information technology – Security techniques – Information Security for Supplier Relationships

- Scope: This international standard covers information security in relationships between acquirers and suppliers to provide appropriate information security management for all parties. In particular, it also includes management of information security risks related to these relationships.

- The standard will be subdivided into the following parts:
  - Part 1 – Overview and Concepts
  - Part 2 – Common Requirements
  - Part 3 – Guidelines for ICT Supply Chain
  - Part 4 – Guidelines for Outsourcing

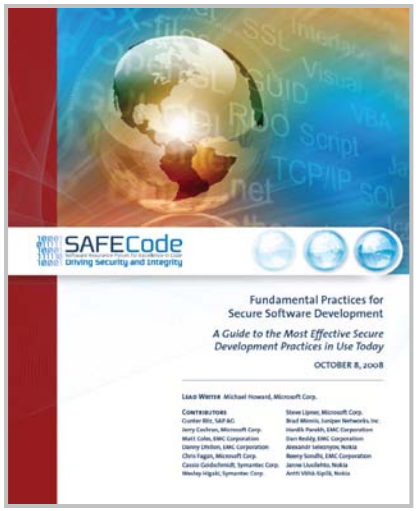
NIST IR 7622, Piloting Supply Chain Risk Management for Federal Information Systems

- Initially based on DoD ICT SCRM Key Practices document and developed in close collaboration with the industry
- Introduces the notion of supply chain players
  - Acquirer - For this document, the acquirer is always a government agency (including those agencies taking on the role of integrator).
  - Integrator – A third-party organization that specializes in combining products/elements of several suppliers to produce elements (information systems).
  - Supplier – Third-party organization providing individual elements. *Synonymous with vendor and manufacturer; also applies to maintenance/disposal service providers*
- Lays out pre-requisites of being able to address ICT SCRM challenge
- States specific practices that are consistent with DoD guidance and ISO frameworks

SAFECode (www.safecode.org)

- SAFECode is a global, industry-led effort to identify and promote best practices for developing and delivering more secure and reliable software, hardware and services

- White papers
  - Software Assurance: An Overview of Current Industry Best Practices
  - Fundamental Practices for Secure Software Development
  - Security Engineering Training: A Framework for Corporate Training Programs on the Principles of Secure Software Development
  - Framework for Software Supply Chain Integrity
  - Software Integrity Controls: An Assurance-Based Approach to Minimizing Risks in the Software Supply Chain



The Open Group
Trusted Technology Provider Framework (TTPF)

Purpose

Identify and gain consensus on common processes, techniques, methods, product and system testing procedures, and language to describe and guide product development and supply chain management practices that can mitigate vulnerabilities which could lead to exploitation and malicious threats to product integrity.

Objectives

- Identify product assurance practices that should be expected from all commercial technology vendors based on the baseline best practices of leading trusted commercial technology suppliers
- Help establish expectations for global government and commercial customers when seeking to identify a trusted technology supplier
- Leverage existing globally recognized information assurance practices and standards
- Share with commercial technology consumers secure manufacturing and trustworthy technology supplier best practices
- Harmonize language used to describe best practices

Source: Source: September 28, 2010 SwA Forum, DoD Trusted Defense Systems, Ms. Kristen Baldwin, DDR&E/Systems Engineering

What's next?

- Continued collaboration to:
  - Reach and enable developers
  - Reach and enable executives
  - Develop and promote resources for us by developers and executives
- Participation in international standardization efforts
  - SC7 TAG intersections through your SC7 TAG
  - CS1/SC27
  - IEEE representative to the SC7 TAG
  - SC22
- Participation through the SwA Working Groups and Forum
- Stay Tuned …

Contact information

Nadya Bartol
Senior Associate

Booz | Allen | Hamilton

Booz Allen Hamilton Inc.
One Preserve Parkway
Rockville, MD 20852
Tel (301) 444-4114
bartol_nadya@bah.com

Michele Moss
Lead Associate

Booz | Allen | Hamilton

Booz Allen Hamilton Inc.
8283 Greensboro Drive
Mclean, VA 22102
Tel (703) 377-1254
moss_michele@bah.com