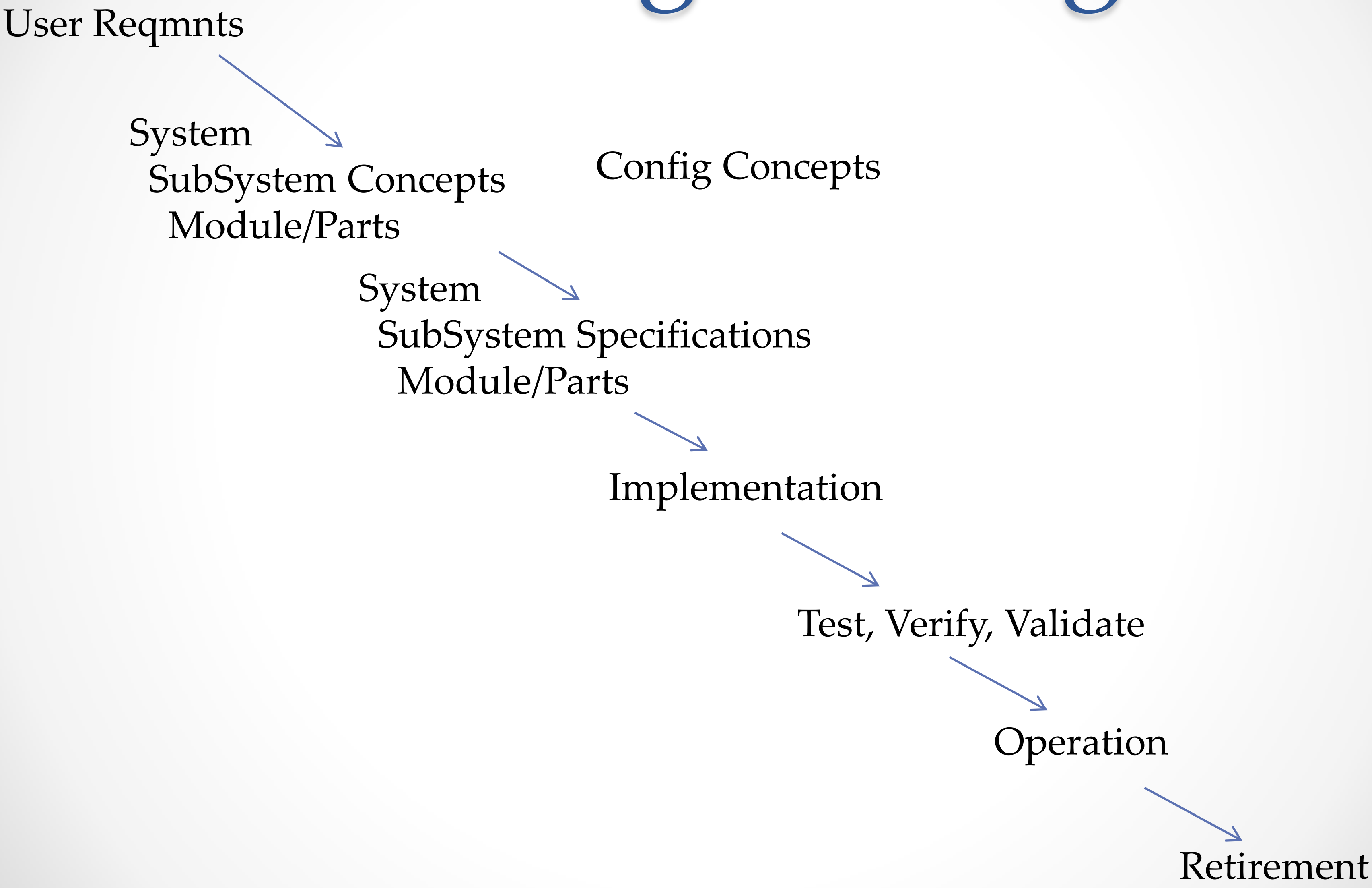
Comments on TR24772

Tatsuaki Takebe
ISO/IEC JTC 1/SC 22
Yokogawa Electric Corp.

Basic Concepts

- Security is not achieved without careful analysis, inspection and efforts.

SW Engineering



User Reqmnts

System
SubSystem Concepts
Module/Parts

Config Concepts

System
SubSystem Specifications
Module/Parts

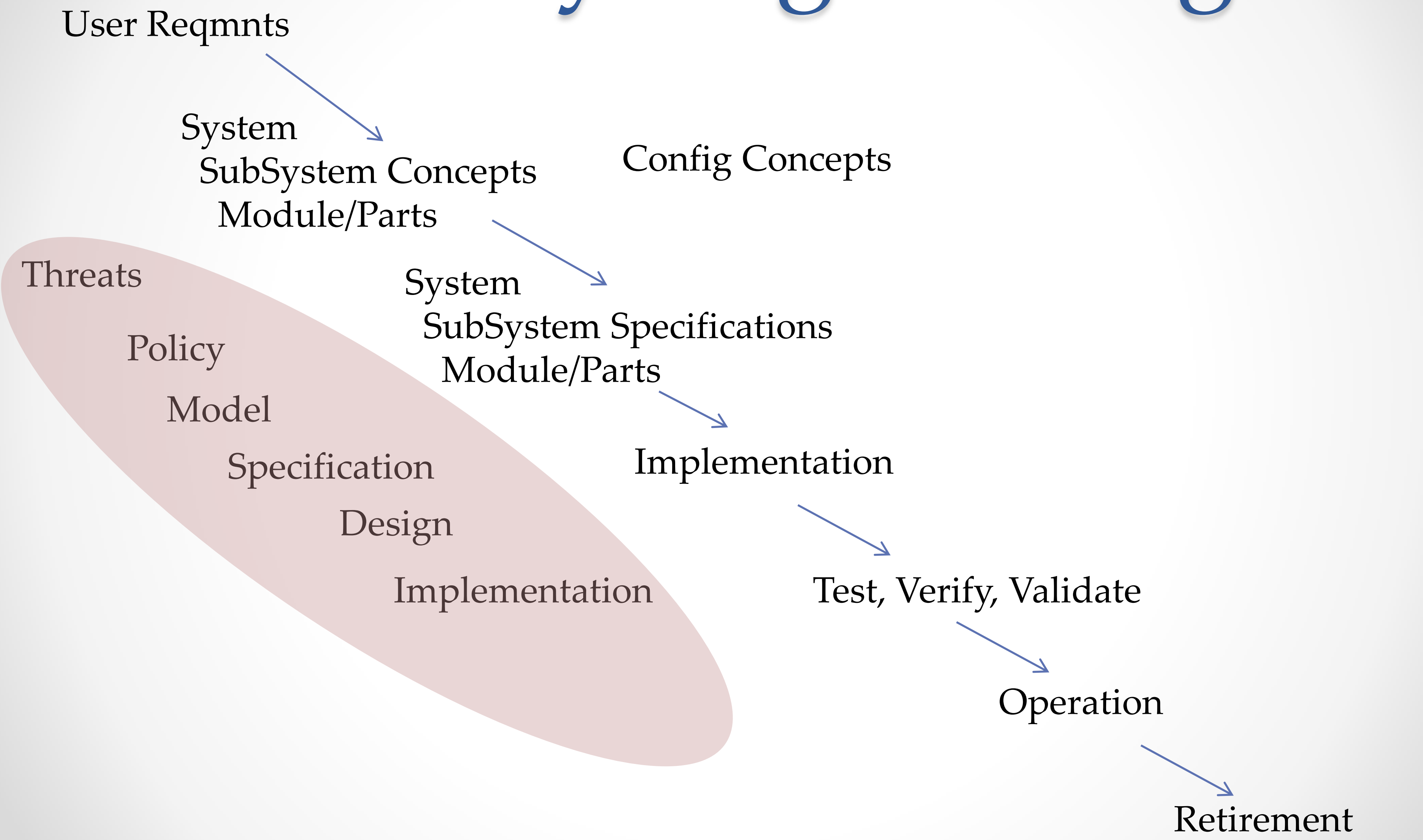
Implementation

Test, Verify, Validate

Operation

Retirement

Security Engineering



User Reqmnts

System
SubSystem Concepts
Module/Parts

Config Concepts

System
SubSystem Specifications
Module/Parts

Implementation

Test, Verify, Validate

Operation

Retirement

Threats

Policy

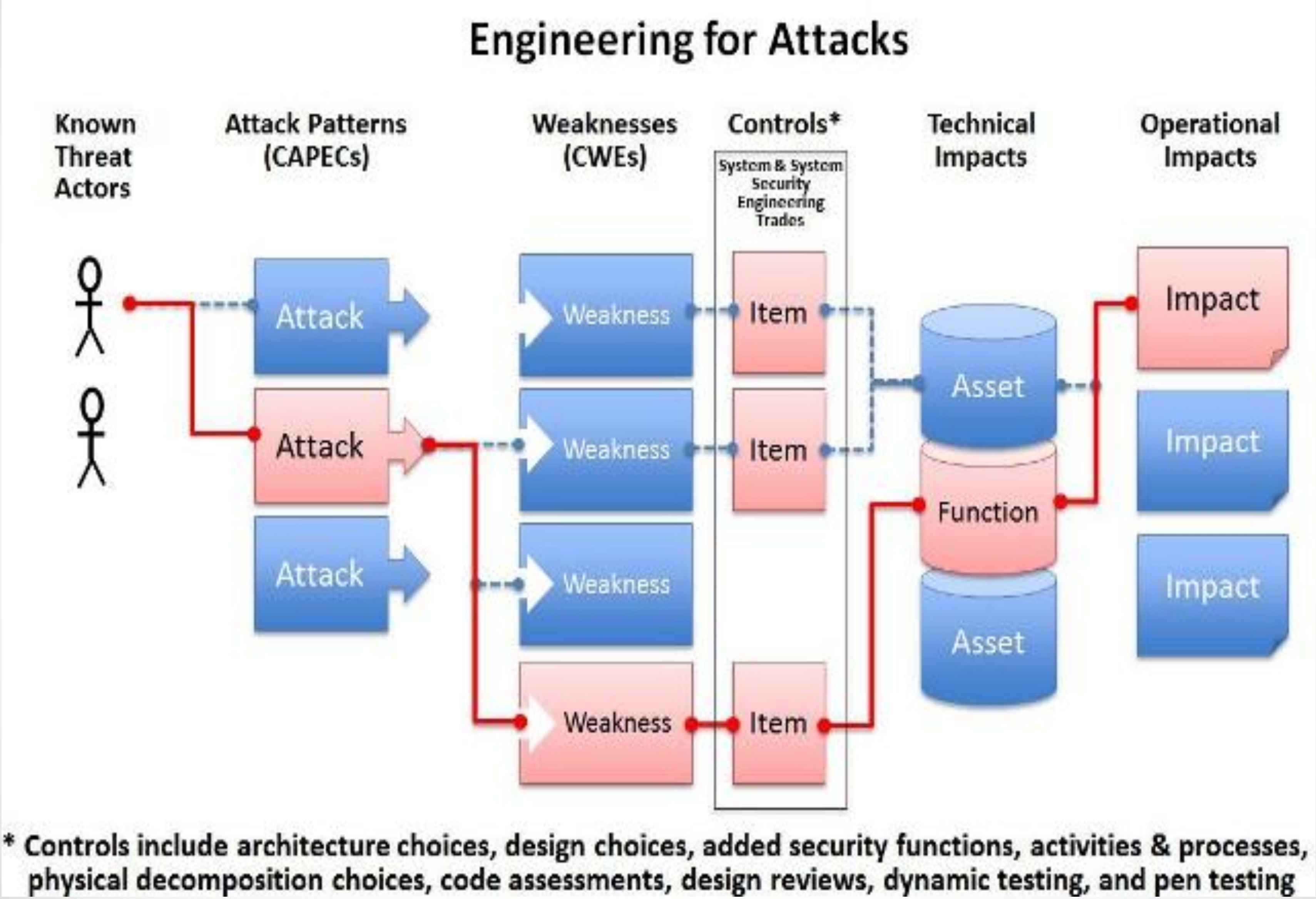
Model

Specification

Design

Implementation

Methodology described in the new ISO/IEC Technical Report 20004, "Refining software vulnerability analysis under ISO/IEC 15408 and ISO/IEC 18045"



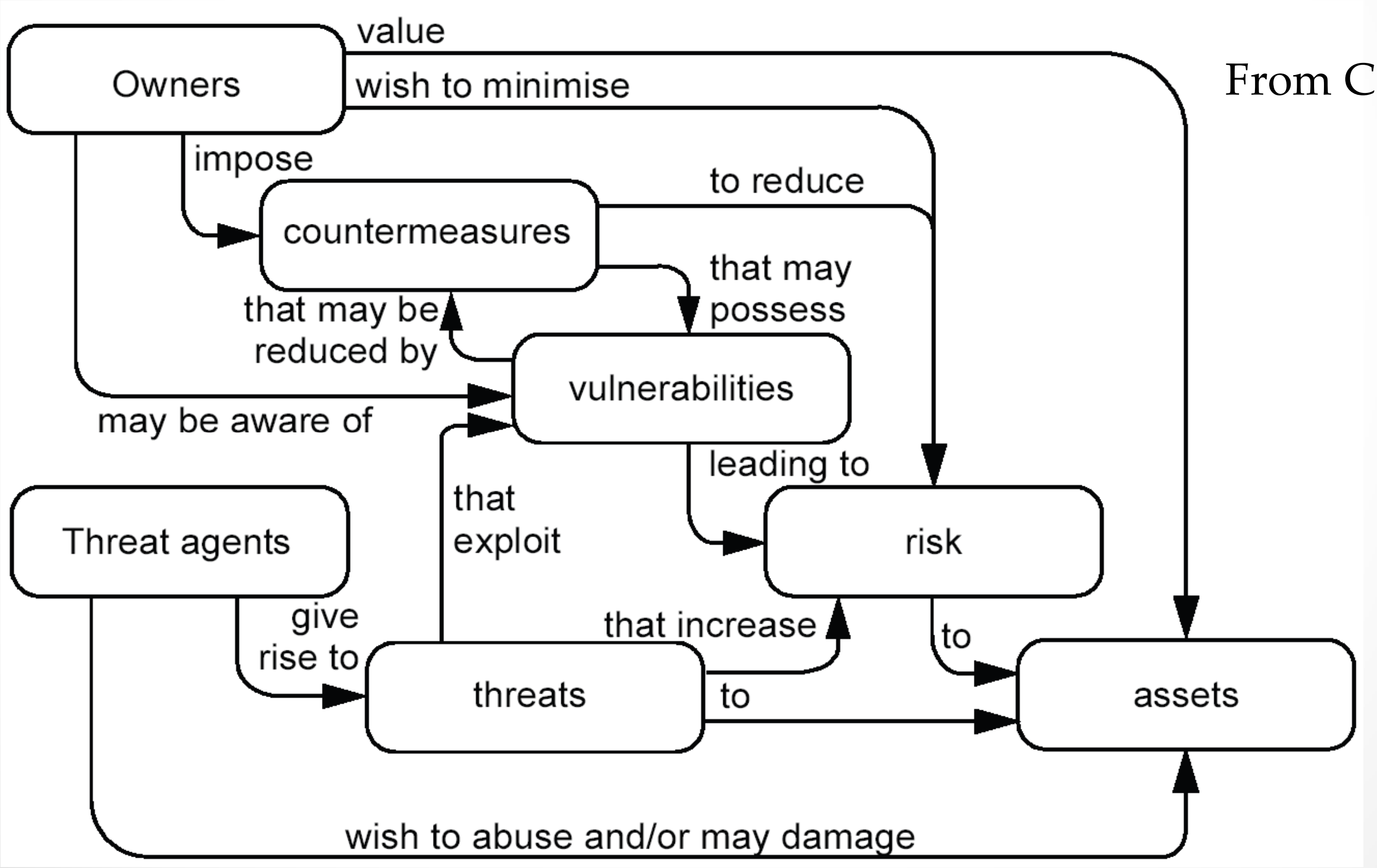
* Controls include architecture choices, design choices, added security functions, activities & processes, physical decomposition choices, code assessments, design reviews, dynamic testing, and pen testing

From Robert Martin

As presented on: http://cwe.mitre.org/community/swa/attacks.html

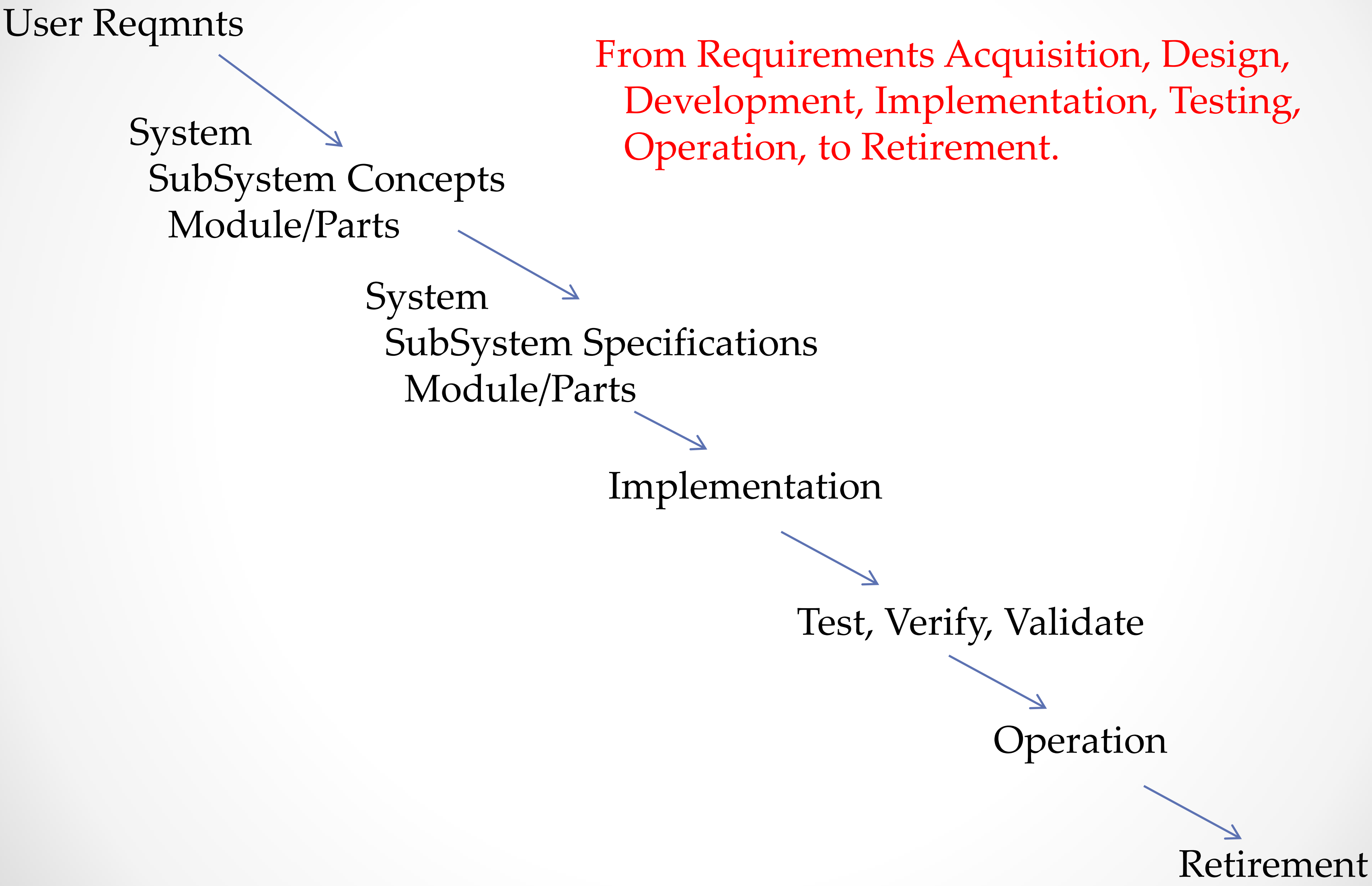
Vulnerabilities in CC 2.3

- Find the vulnerabilities and provide countermeasures until the residual risk is acceptable.



From CC Part 1 V2.3

Potential Vulnerabilities reside in every process.



User Reqmnts

System
SubSystem Concepts
Module/Parts

System
SubSystem Specifications
Module/Parts

Implementation

Test, Verify, Validate

Operation

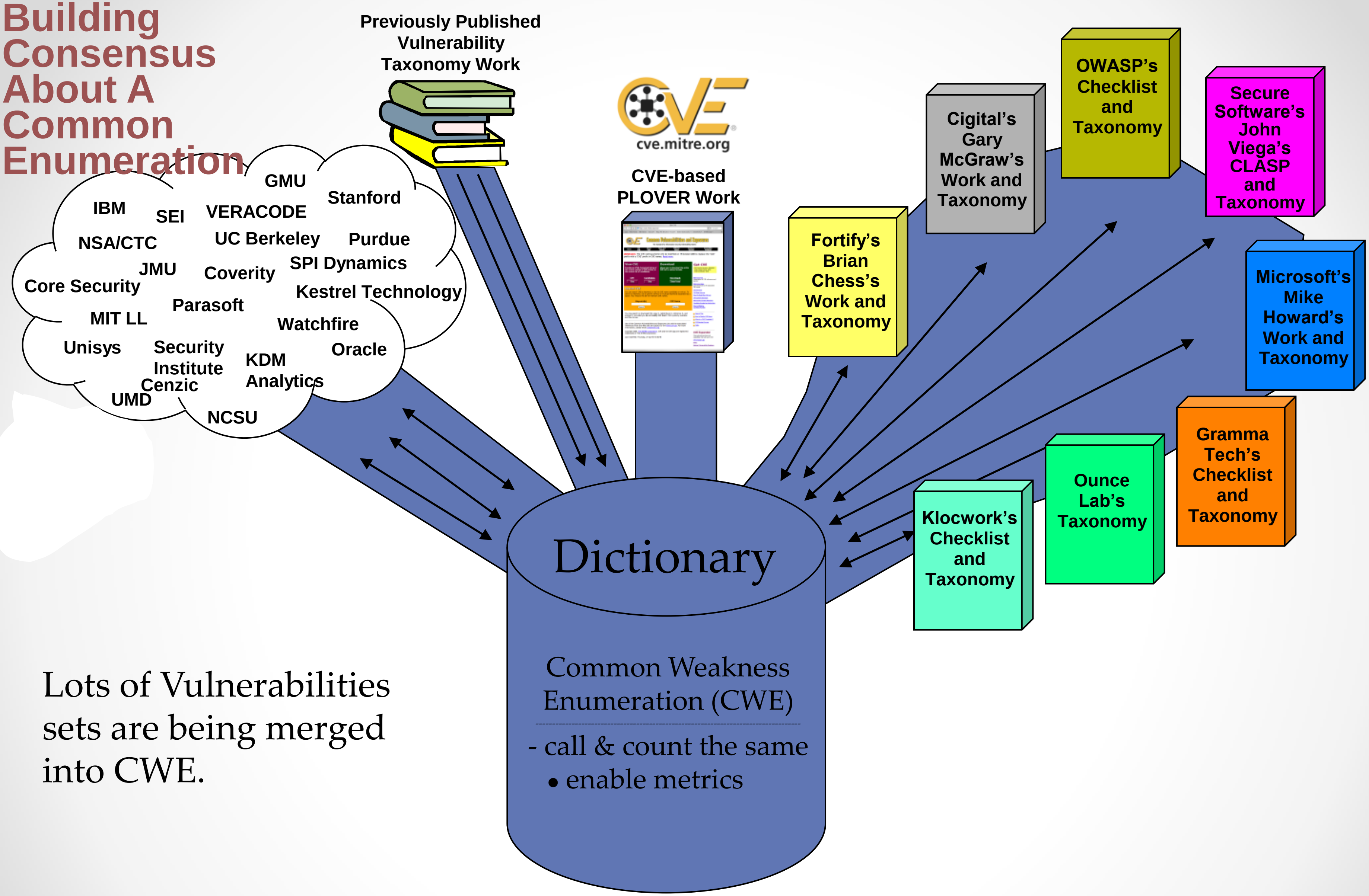
Retirement

From Requirements Acquisition, Design, Development, Implementation, Testing, Operation, to Retirement.

The Key Issues are Vulnerabilities

- How to find them?
- How to manage the risk caused by the vulnerabilities?
- If the vulnerabilities are studied, classified, published, and maintained, this will probably make the life easier for the vendors and the asset owners.
- Are there such databases?
- CWE. And several.

Building Consensus About A Common Enumeration


Previously Published Vulnerability Taxonomy Work
cve.mitre.org
CVE-based PLOVER Work
IBM
SEI
VERACODE
GMU
Stanford
NSA/CTC
UC Berkeley
Purdue
JMU
Coverity
SPI Dynamics
Core Security
Kestrel Technology
Parasoft
MIT LL
Watchfire
Unisys
Security Institute
KDM Analytics
Oracle
Cenzic
UMD
NCSU
Cigital's Gary McGraw's Work and Taxonomy
OWASP's Checklist and Taxonomy
Secure Software's John Viega's CLASP and Taxonomy
Fortify's Brian Chess's Work and Taxonomy
Microsoft's Mike Howard's Work and Taxonomy
Gramma Tech's Checklist and Taxonomy
Ounce Lab's Taxonomy
Klocwork's Checklist and Taxonomy
Dictionary

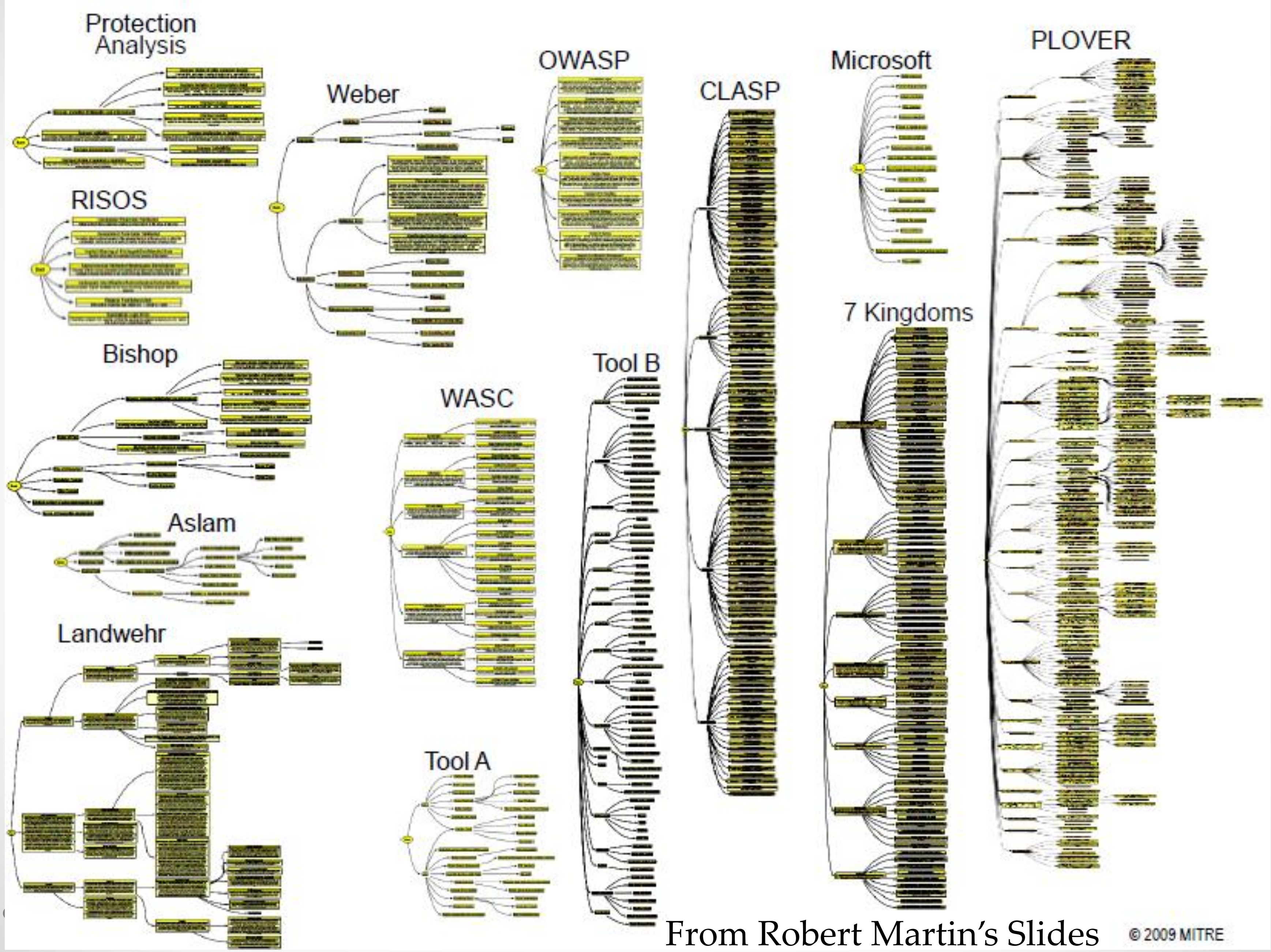
Common Weakness Enumeration (CWE)

- call & count the same
• enable metrics

Lots of Vulnerabilities sets are being merged into CWE.

From Robert Martin's Slides

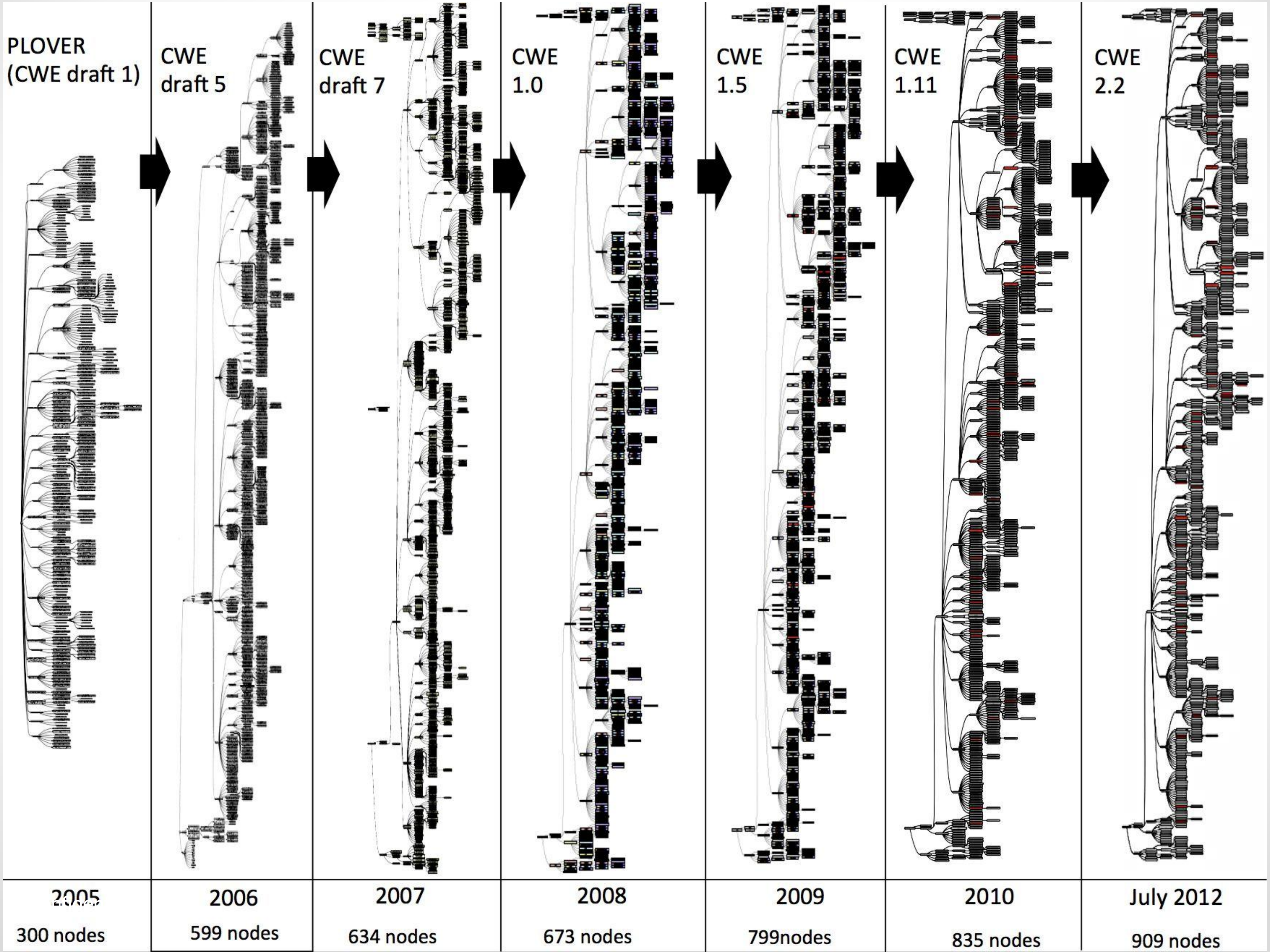
Sources for CWE



From Robert Martin's Slides

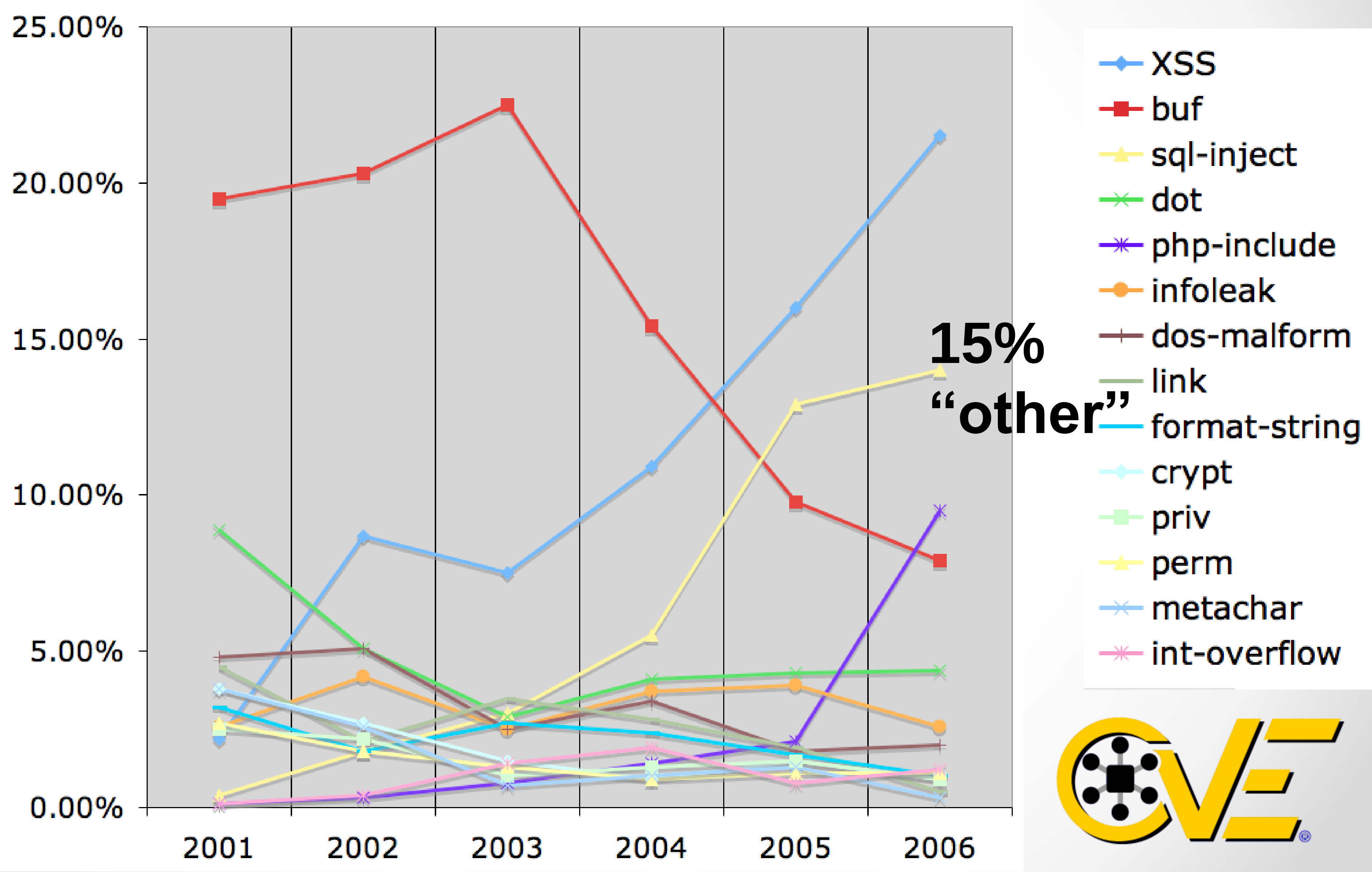
CWE growth

From Robert Martin's Slides



From Robert Martin’s Slides

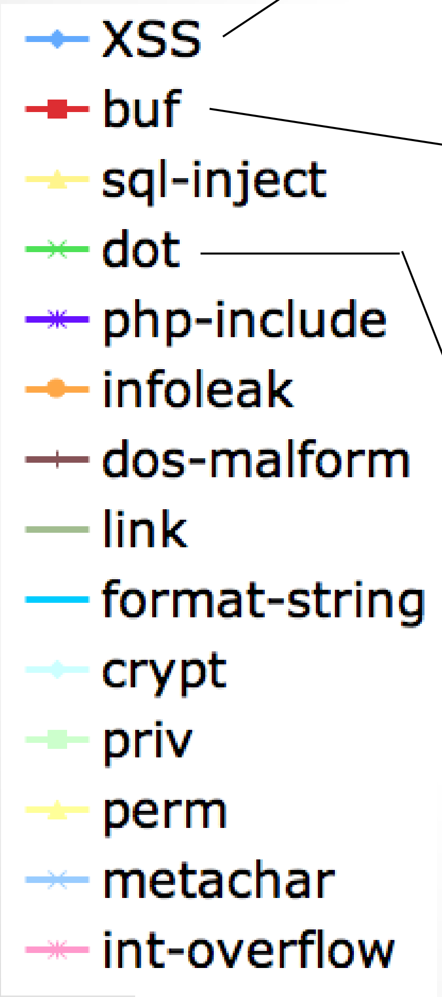
Vulnerability Type Trends: A Look at the CVE List (2001 - 2006)



15% “other”

Removing and Preventing the Vulnerabilities Requires More Specific Definitions…

From Robert Martin’s Slides



Cross-site scripting (XSS):

- Basic XSS
- XSS in error pages
- Script in IMG tags
- XSS using Script in Attributes
- XSS using Script Via Encoded URI Schemes
- Doubled character XSS manipulations, e.g. '<<script'
- Invalid Characters in Identifiers
- Alternate XSS syntax

Buffer Errors

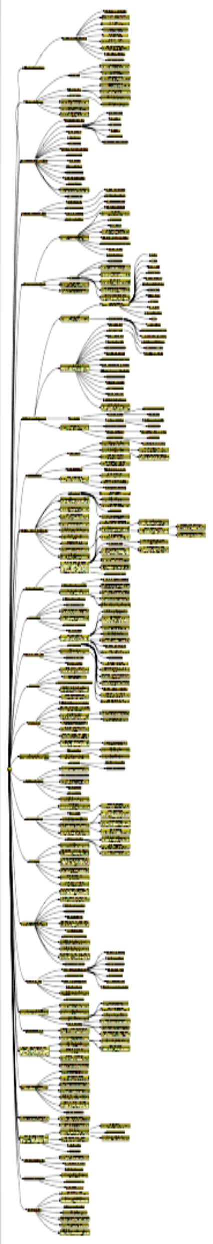
- Unbounded Transfer ('classic overflow')
- Write-what-where condition
- Boundary beginning violation ('buffer underwrite')
- Out-of-bounds Read
- Wrap-around error
- Unchecked array indexing
- Length Parameter Inconsistency
- Other length calculation error
- Miscalculated null termination
- String Errors

Relative Path Traversal

- Path Issue - dot dot slash - '../filedir'
- Path Issue - leading dot dot slash - '/../filedir'
- Path Issue - leading directory dot dot slash - '/directory/../filename'
- Path Issue - directory doubled dot dot slash - 'directory/../../filename'
- Path Issue - dot dot backslash - '..¥filename'
- Path Issue - leading dot dot backslash - '¥..¥filename'
- Path Issue - leading directory dot dot backslash - '¥directory¥..¥filename'
- Path Issue - directory doubled dot dot backslash - 'directory¥..¥..¥filename'
- Path Issue - triple dot - '...'
- Path Issue - multiple dot - '....'
- Path Issue - doubled dot dot slash - '....//'
- Path Issue - doubled triple dot slash - '.../...//'

From Robert Martin’s Slides

PLOVER:
300 “types”of Weaknesses, 1500 real-world CVE examples



[BUFF] Buffer overflows, format strings, etc.	10 types
[SVM] Structure and Validity Problems	10 types
[SPEC] Special Elements (Characters or Reserved Words)	19 types
[SPECM] Common Special Element Manipulations	11 types
[SPECTS] Technology-Specific Special Elements	17 types
[PATH] Pathname Traversal and Equivalence Errors	47 types
[CP] Channel and Path Errors	13 types
[CCC] Cleansing, Canonicalization, and Comparison Errors	16 types
[INFO] Information Management Errors	19 types
[RACE] Race Conditions	6 types
[PPA] Permissions, Privileges, and ACLs	20 types
[HAND] Handler Errors	4 types
[UI] User Interface Errors	7 types
[INT] Interaction Errors	7 types
[INIT] Initialization and Cleanup Errors	6 types
[RES] Resource Management Errors	11 types
[NUM] Numeric Errors	6 types
[AUTHENT] Authentication Error	12 types
[CRYPTO] Cryptographic errors	13 types
[RAND] Randomness and Predictability	9 types
[CODE] Code Evaluation and Injection	4 types
[ERS] Error Conditions, Return Values, Status Codes	4 types
[VER] Insufficient Verification of Data	7 types
[MAID] Modification of Assumed-Immutable Data	2 types
[MAL] Product-Embedded Malicious Code	7 types
[ATTMIT] Common Attack Mitigation Failures	3 types
[CONT] Containment errors (container errors)	3 types
[MISC] Miscellaneous WIFFs	7 types

From Robert Martin’s Slides

Goal of the Common Weakness Enumeration Initiative

- To improve the quality of software with respect to known security issues within source code
    - define a unified measurable set of weaknesses
    - enable more effective discussion, description, selection and use of software security tools and services that can find these weaknesses

From Robert Martin’s Slides

Clarifying software weaknesses: Enabling communication (1 of 2)

- Systems Development Manager Issue Areas:
  - What are the software weaknesses I need to protect against
    - Architecture, design, code
  - Can I look through the issues by technologies, risks, severity
  - What have the pieces of my system been vetted for?
    - COTS packages, organic development, open source
  - Identify tools to vet code based on tool coverage
    - How effective are the tools?

- Assessment Tool Vendors Issue Areas:
  - Express what my tool does
  - Succinctly identify areas I should expand coverage

From Robert Martin’s Slides

Clarifying software weaknesses:

Enabling communication (2 of 2)

- COTS Product Vendor Issue Areas:
  - What have I vetted my applications for?
  - What do my customers want me to vet for?
- Researcher Issue Areas:
  - Quickly understand what is known
  - Easily identify areas to contribute/refine/correct
- Educator Issue Areas:
  - Train students with the same concepts they’ll use in practice
- Operations Manager Issue Areas:
  - What issues have my applications been vetted for? (COTS/Organic/OS)
  - What types of issues are more critical for my technology?
  - What types of issues are more likely to be successfully exploited?

… which led to the Preliminary List of Vulnerability Examples for Researchers (PLOVER)

- Initial goal: extend vulnerability auditing checklist
- Collected extensive CVE examples
  - Emphasis on 2005 and 2006
  - Reviewed all issues flagged "other“
- 300 weakness types, 1500 real-world CVE examples
- Identified classification difficulties
  - Primary vs. resultant vulns
  - Multi-factor issues
  - Uncategorized examples
  - Tried to separate attacks from vulnerabilities
- Beginning vulnerability theory
  - Properties
  - Manipulations
  - Consequences

From Robert Martin’s Slides

- One of the 3 major sources of CWE

Vulnerability Theory: Problem Statement and Rationale

- With 600+ variants, what are the main themes?
- Why is it so hard to classify vulnerabilities cleanly?
  - CWE, Pernicious Kingdoms, OWASP, others have had similar difficulties
- Same terminology used in multiple dimensions
  - Frequent mix of attacks, threats, weaknesses/faults, consequences
  - E.g. buffer overflows, directory traversal

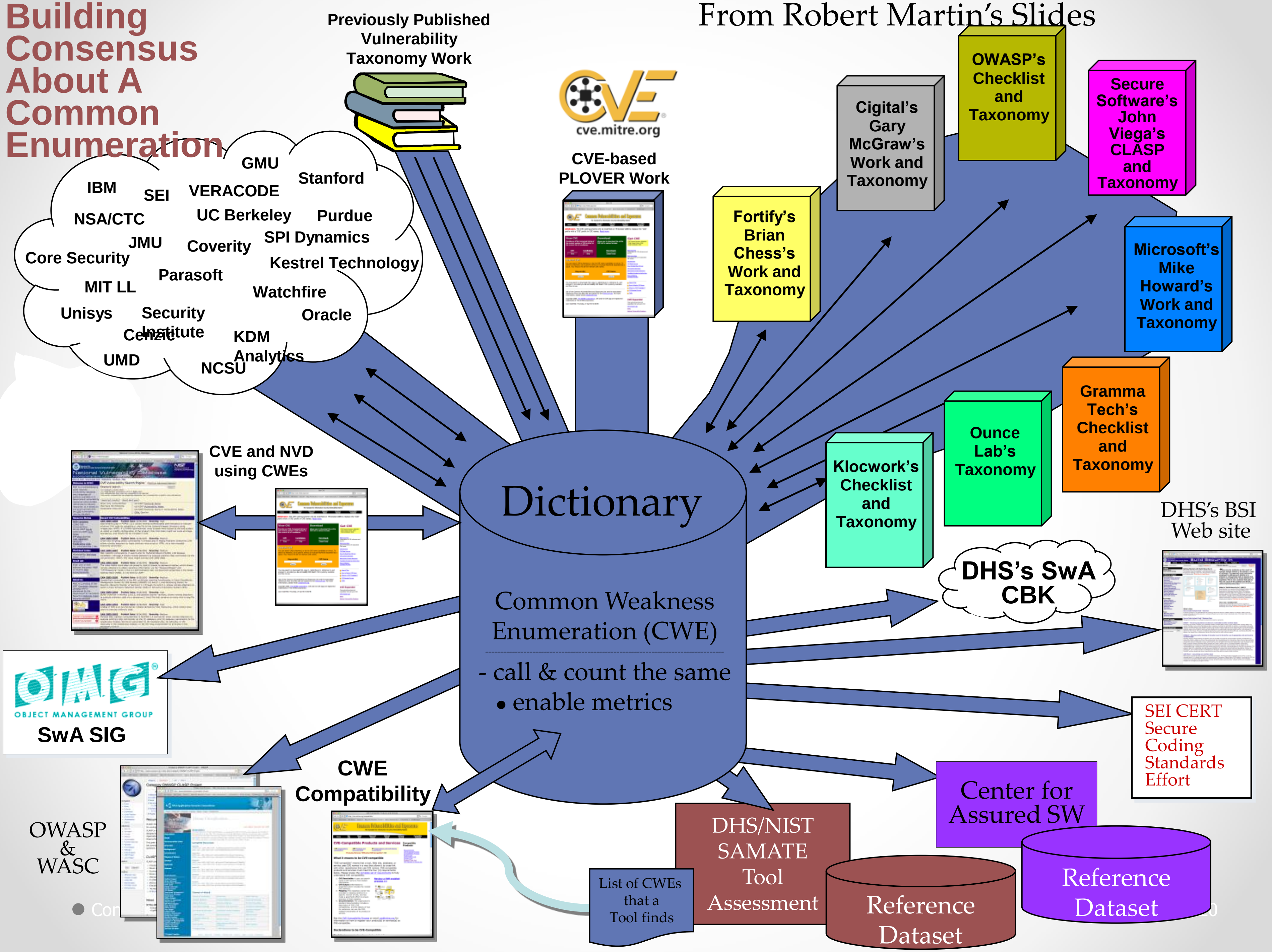
- Goal: Increase understanding of vulnerabilities
  - Vocabulary for more precise discussion
  - Label current inconsistencies in terminology and taxonomy
  - Codify some of the researchers' instinct

- One possible application: gap analysis, defense, and design recommendations
  - "Algorithms X and Y both assume input has property P. Attack pattern A manipulates P to compromise X. Would A succeed against Y?"
  - "Technology Z has properties P1 and P2. What vulnerability classes are most likely to be present?"
  - "Why is XSS so obvious but so hard to eradicate?"

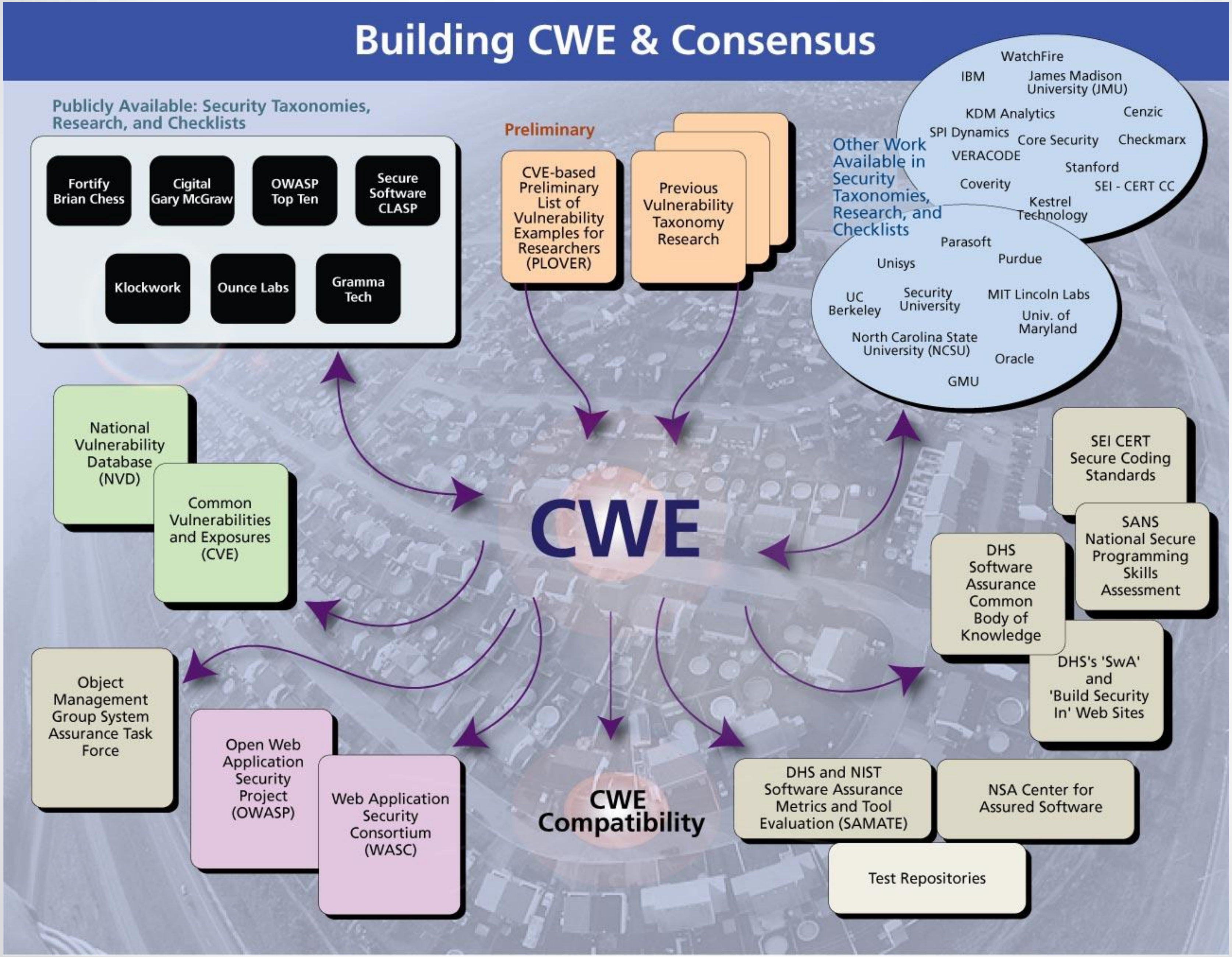
From Robert Martin's Slides

Building Consensus About A Common Enumeration

From Robert Martin's Slides


Previously Published Vulnerability Taxonomy Work
cve.mitre.org
CVE-based PLOVER Work
Cigital's Gary McGraw's Work and Taxonomy
OWASP's Checklist and Taxonomy
Secure Software's John Viega's CLASP and Taxonomy
Fortify's Brian Chess's Work and Taxonomy
Microsoft's Mike Howard's Work and Taxonomy
Gramma Tech's Checklist and Taxonomy
Ounce Lab's Taxonomy
Klocwork's Checklist and Taxonomy
GMU
Stanford
IBM
SEI
VERACODE
NSA/CTC
UC Berkeley
Purdue
JMU
Coverity
SPI Dynamics
Core Security
Kestrel Technology
Parasoft
MIT LL
Watchfire
Unisys
Security Institute
Oracle
Cenzic
KDM Analytics
UMD
NCSU
CVE and NVD using CWEs
Dictionary
Common Weakness Enumeration (CWE)
- call & count the same
• enable metrics
DHS's BSI Web site
DHS's SwA CBK
OBJECT MANAGEMENT GROUP
SwA SIG
SEI CERT Secure Coding Standards Effort
CWE Compatibility
OWASP & WASC
List of CWEs that a Tool finds
DHS/NIST SAMATE Tool Assessment
Reference Dataset
Center for Assured SW
Reference Dataset

From Robert Martin's Slides



Building CWE & Consensus

Publicly Available: Security Taxonomies, Research, and Checklists

- Fortify Brian Chess
- Cigital Gary McGraw
- OWASP Top Ten
- Secure Software CLASP
- Klockwork
- Ounce Labs
- Gramma Tech

Preliminary

- CVE-based Preliminary List of Vulnerability Examples for Researchers (PLOVER)
- Previous Vulnerability Taxonomy Research

Other Work Available in Security Taxonomies, Research, and Checklists

- WatchFire
- IBM
- James Madison University (JMU)
- KDM Analytics
- Cenzic
- SPI Dynamics
- Core Security
- Checkmarx
- VERACODE
- Stanford
- Coverity
- SEI - CERT CC
- Kestrel Technology
- Parasoft
- Unisys
- Purdue
- UC Berkeley
- Security University
- MIT Lincoln Labs
- Univ. of Maryland
- North Carolina State University (NCSU)
- Oracle
- GMU

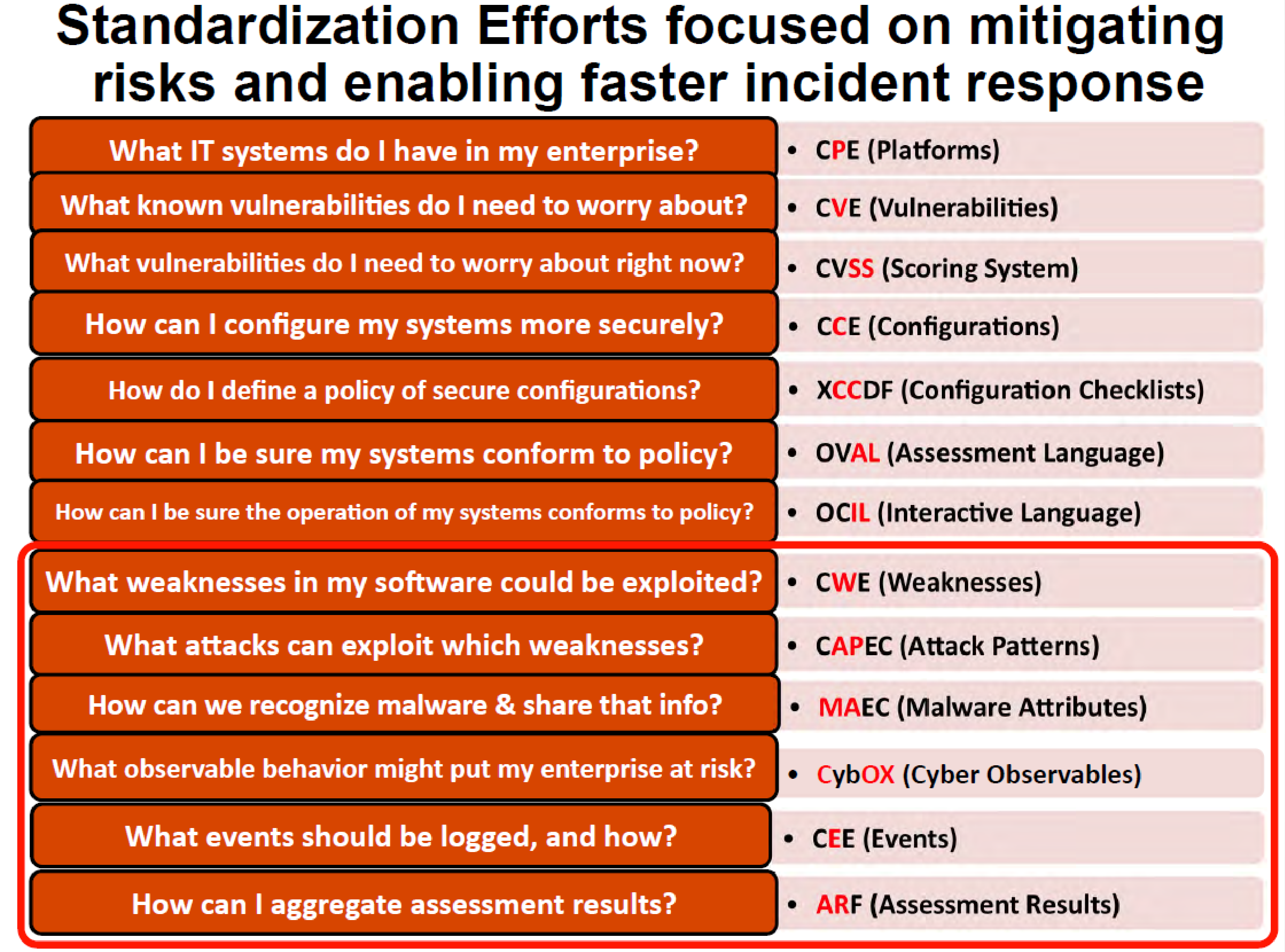
CWE

- National Vulnerability Database (NVD)
- Common Vulnerabilities and Exposures (CVE)
- SEI CERT Secure Coding Standards
- SANS National Secure Programming Skills Assessment
- DHS Software Assurance Common Body of Knowledge
- DHS's 'SwA' and 'Build Security In' Web Sites
- Object Management Group System Assurance Task Force
- Open Web Application Security Project (OWASP)
- Web Application Security Consortium (WASC)

CWE Compatibility

- DHS and NIST Software Assurance Metrics and Tool Evaluation (SAMATE)
- NSA Center for Assured Software
- Test Repositories

From Robert Martin's Slides



Standardization Efforts focused on mitigating risks and enabling faster incident response

What IT systems do I have in my enterprise?	• CPE (Platforms)
What known vulnerabilities do I need to worry about?	• CVE (Vulnerabilities)
What vulnerabilities do I need to worry about right now?	• CVSS (Scoring System)
How can I configure my systems more securely?	• CCE (Configurations)
How do I define a policy of secure configurations?	• XCCDF (Configuration Checklists)
How can I be sure my systems conform to policy?	• OVAL (Assessment Language)
How can I be sure the operation of my systems conforms to policy?	• OCIL (Interactive Language)
What weaknesses in my software could be exploited?	• CWE (Weaknesses)
What attacks can exploit which weaknesses?	• CAPEC (Attack Patterns)
How can we recognize malware & share that info?	• MAEC (Malware Attributes)
What observable behavior might put my enterprise at risk?	• CybOX (Cyber Observables)
What events should be logged, and how?	• CEE (Events)
How can I aggregate assessment results?	• ARF (Assessment Results)

- Reference to CWE means you can get other related information.

Proposed procedures

- Longterm Procedure
  - Review CWE and identify programming language related CWE element.
  - Review Sub-clauses 6.3, 6.4, …., 6.57 and identify those without CWE cross reference. Try to find CWE using keywords from 6.*. This will find appropriate CWE references.
  - Review chapter 7, chapter 8 to find the clauses without CWE cross references. Try to find CWE using keywords.
- This time (proposed comment @Sep 2012)
  - Look into CWE top 25.
  - Find uncovered CWE.
  - Try to find the reference slots where we can put uncovered CWE.
  - Try to think what we can do with the still not covered CWE.