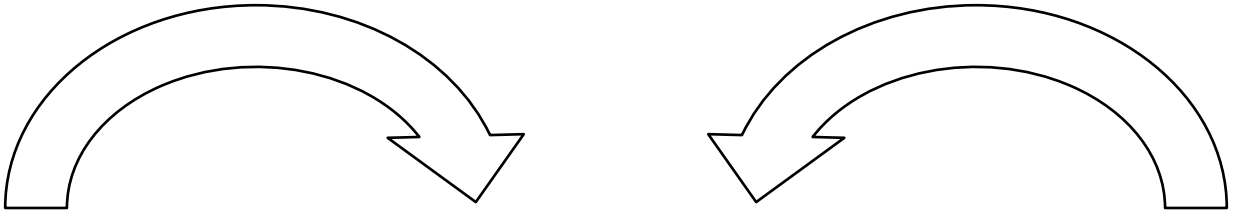
Risk Analysis and Measurement with CWRAF

- Common Weakness Risk Analysis Framework -

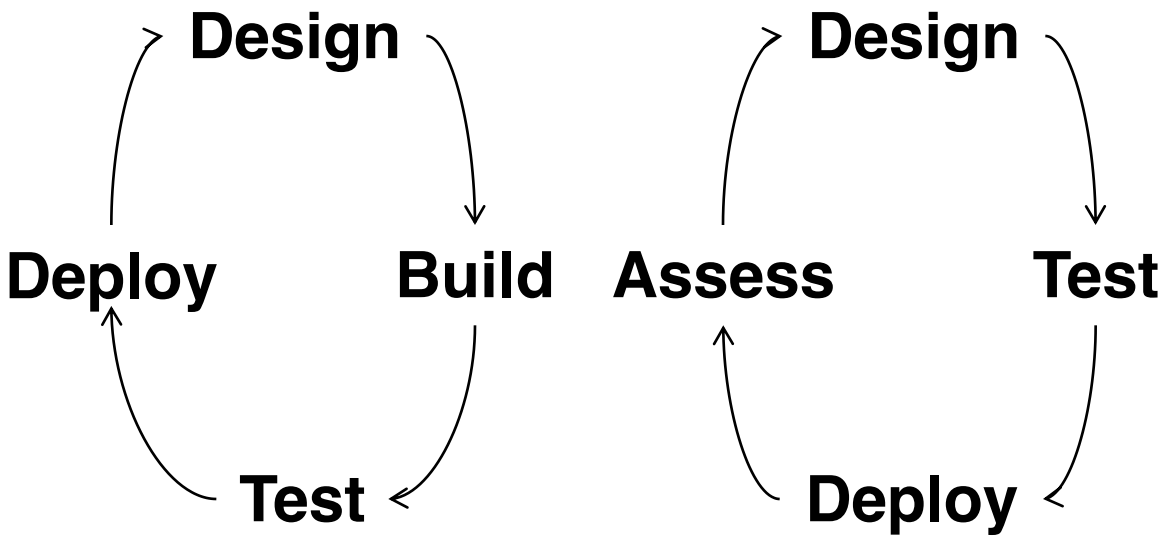
April 4, 2012

Making Security Measurable (MSM)

"You Are Here"



Software Assurance | **Enterprise Security Management** | **Threat Management**



- Vulnerabilities
- Exploits
- Attacks
- Malware

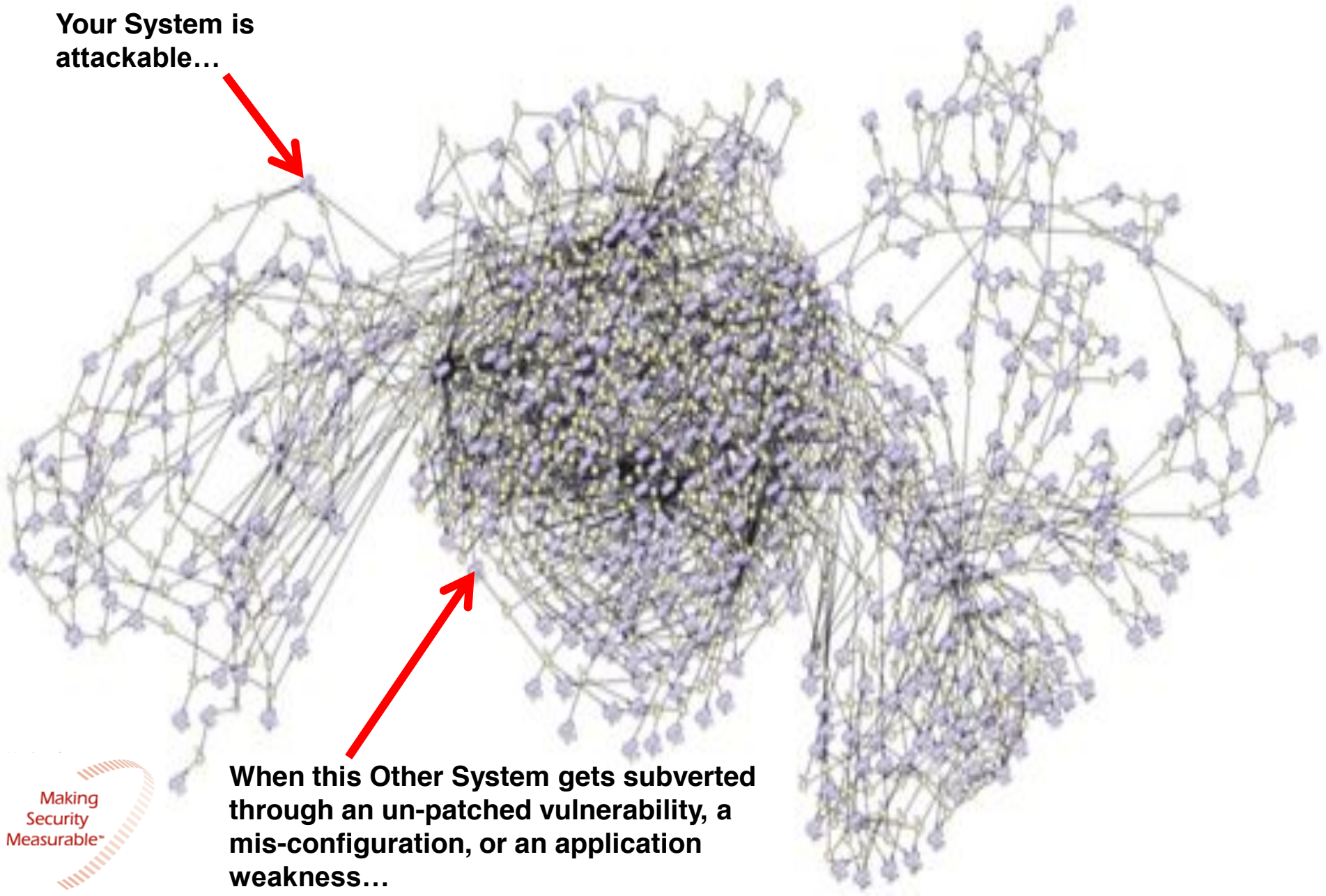
CWE, CAPEC, CWSS, CWRAF

CPE, CCE, OVAL, OCIL, XCCDF, AssetId, ARF

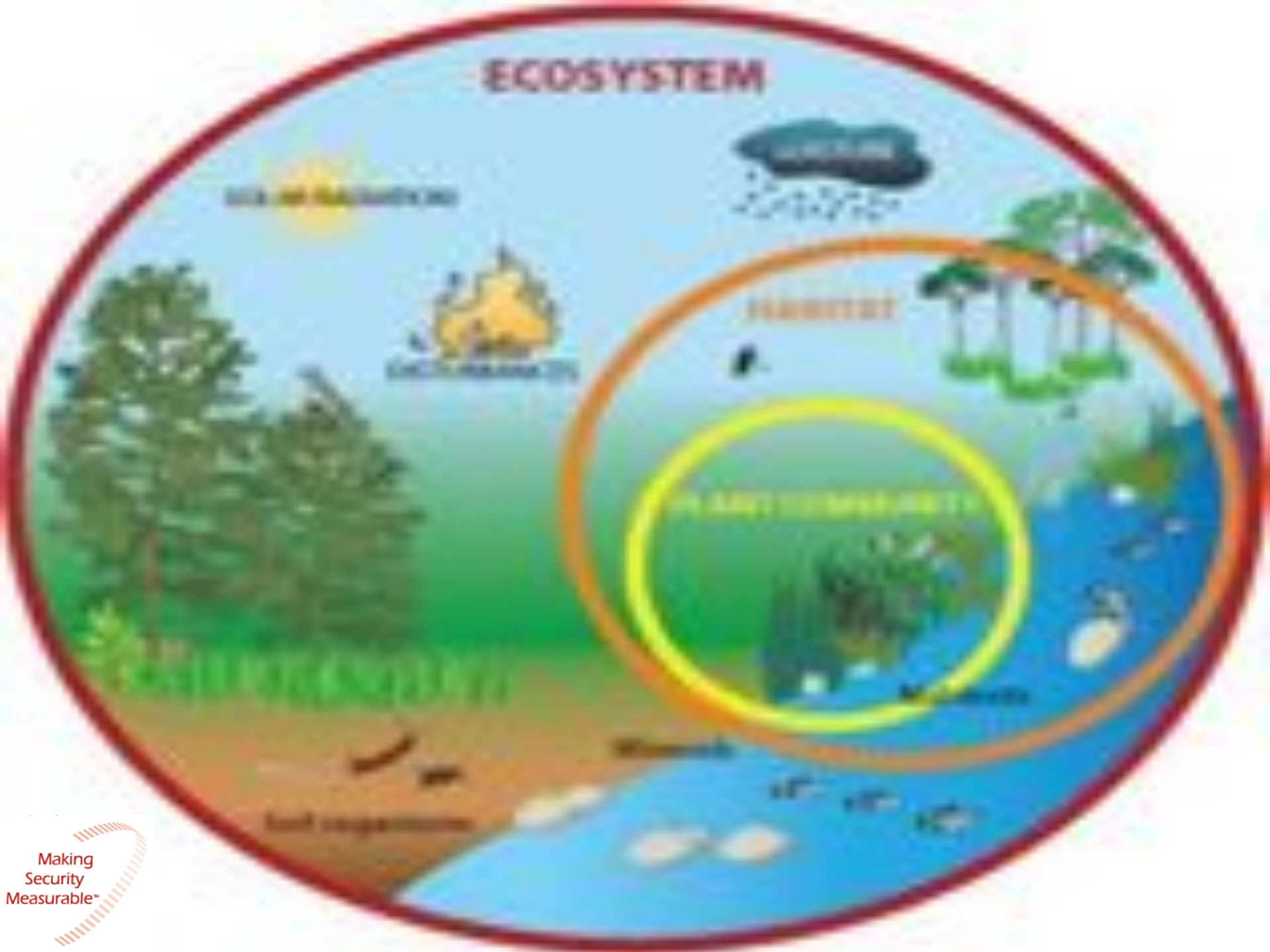
CVE, CWE, CAPEC, MAEC, CybOX, IODEF, RID, RID-T, CYBEX

© 2012 MITRE

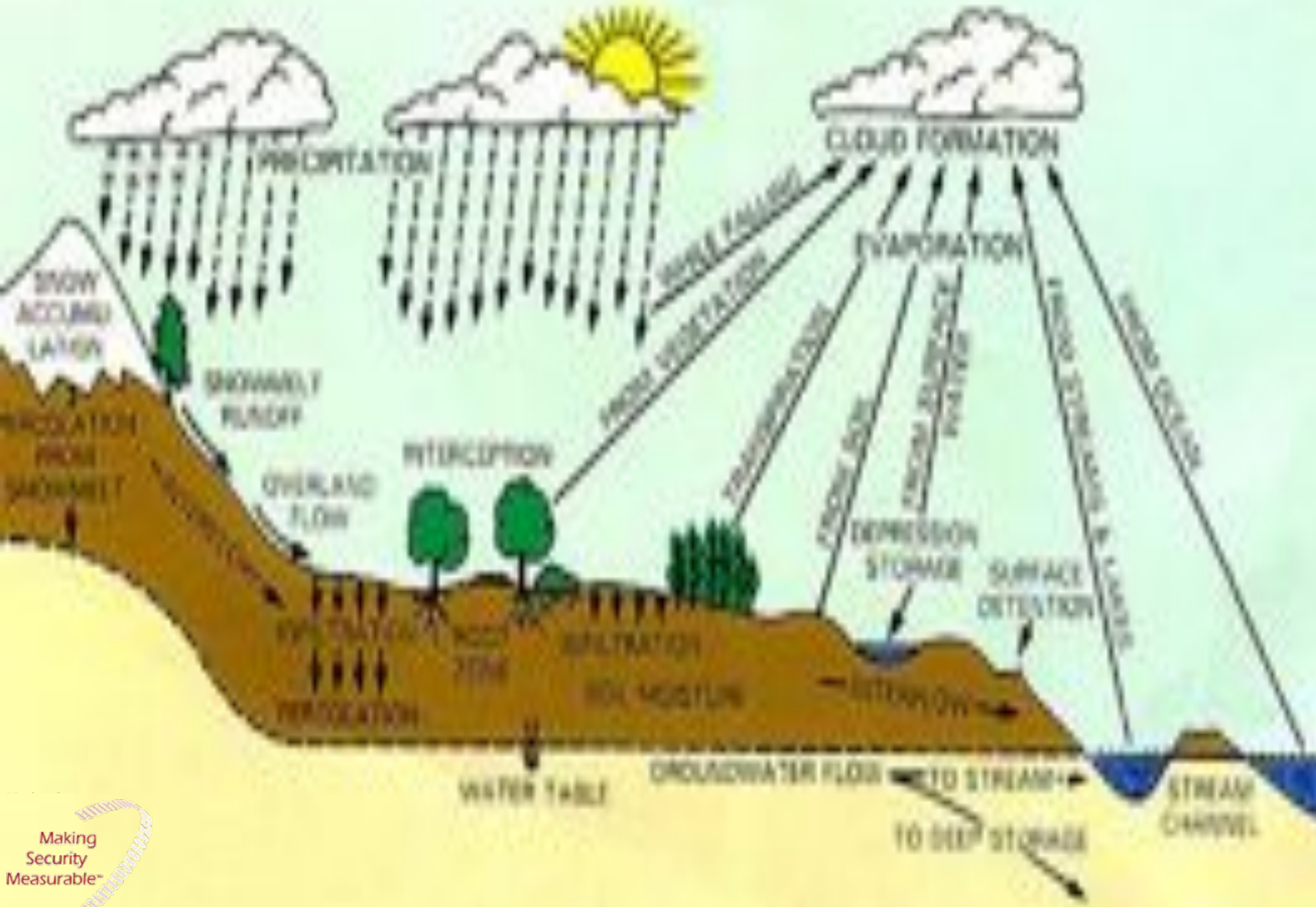
Today Everything's Connected – Like an Ecosystem



Making Security Measurable™

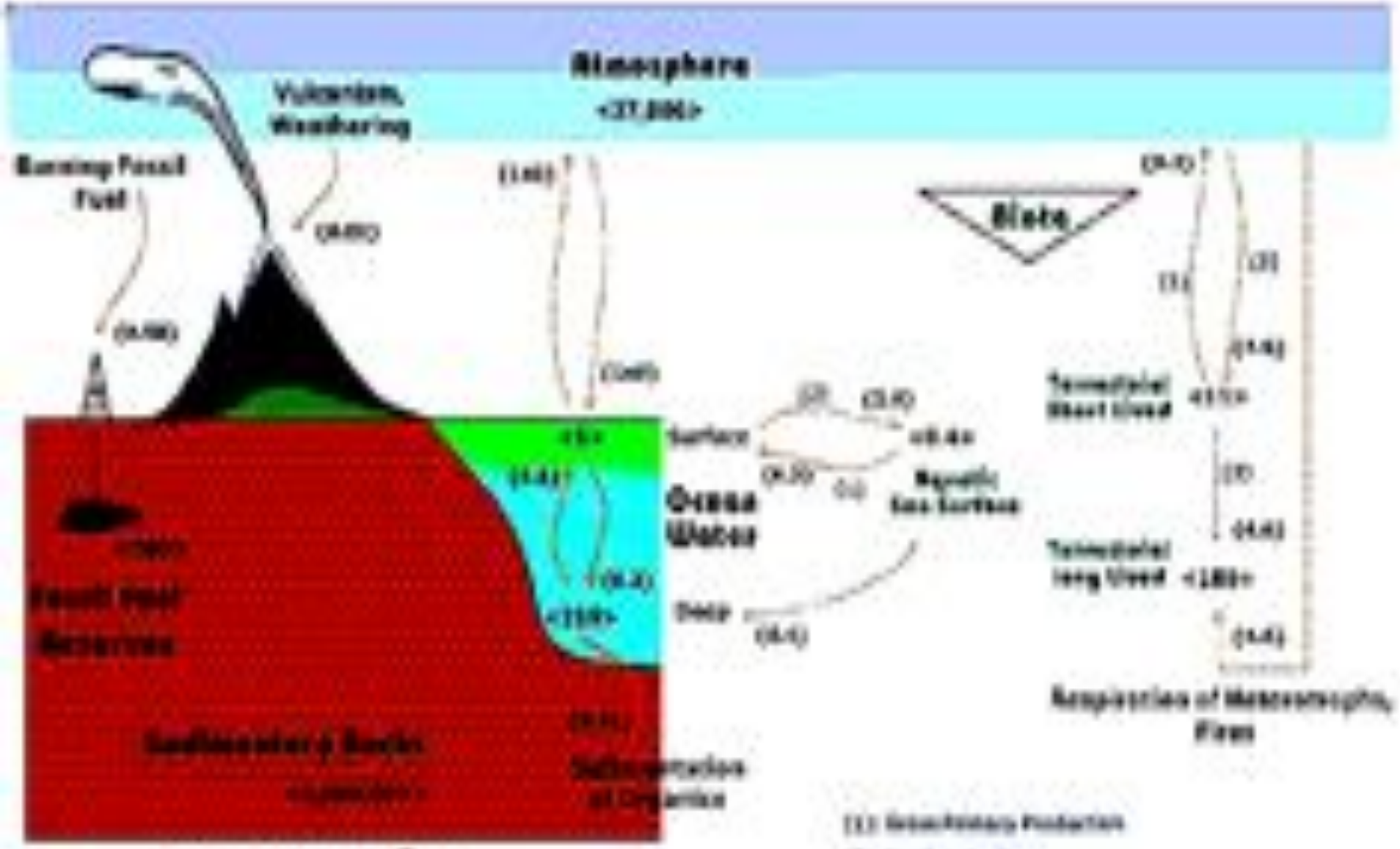


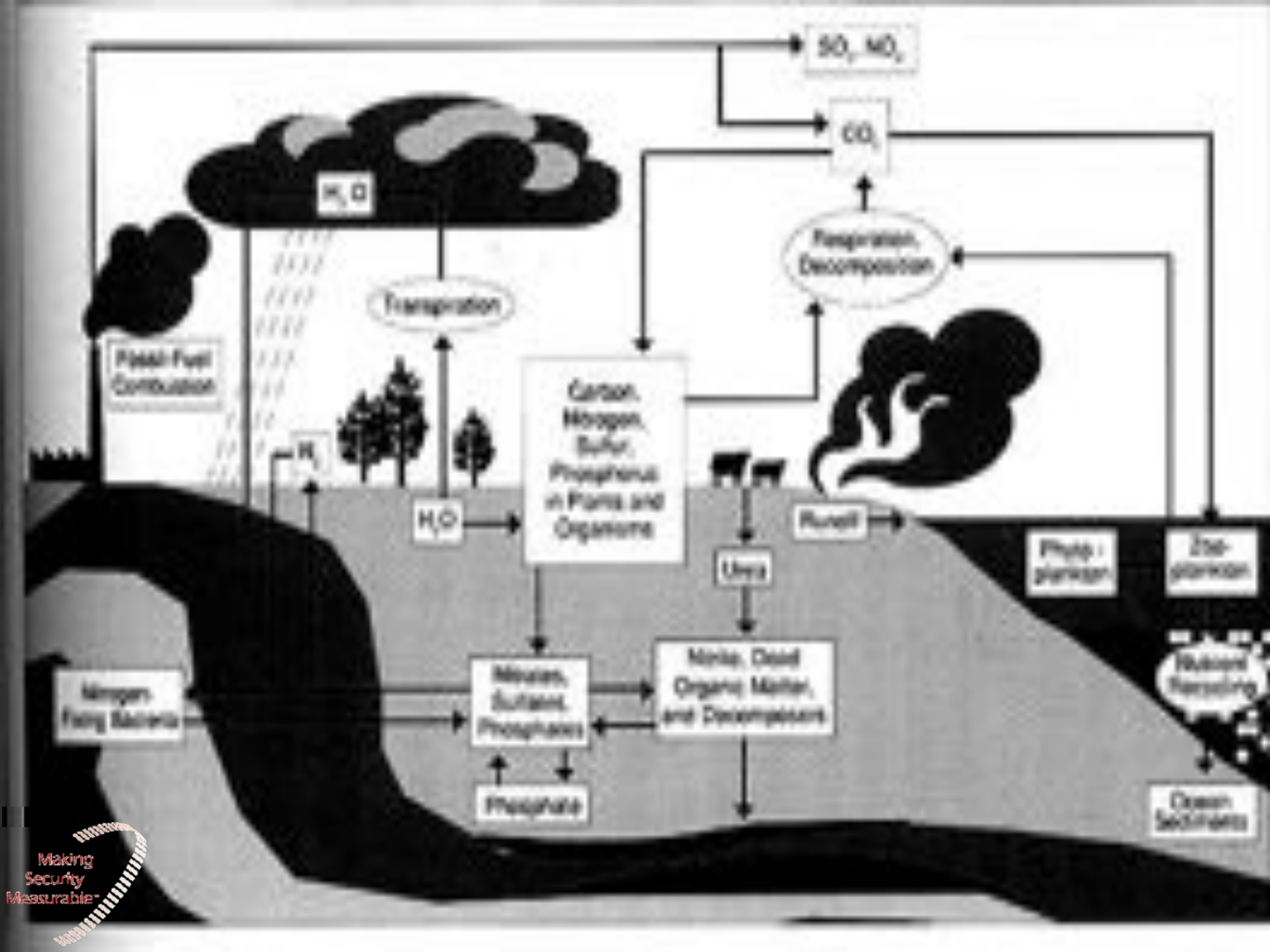
Making Security Measurable™



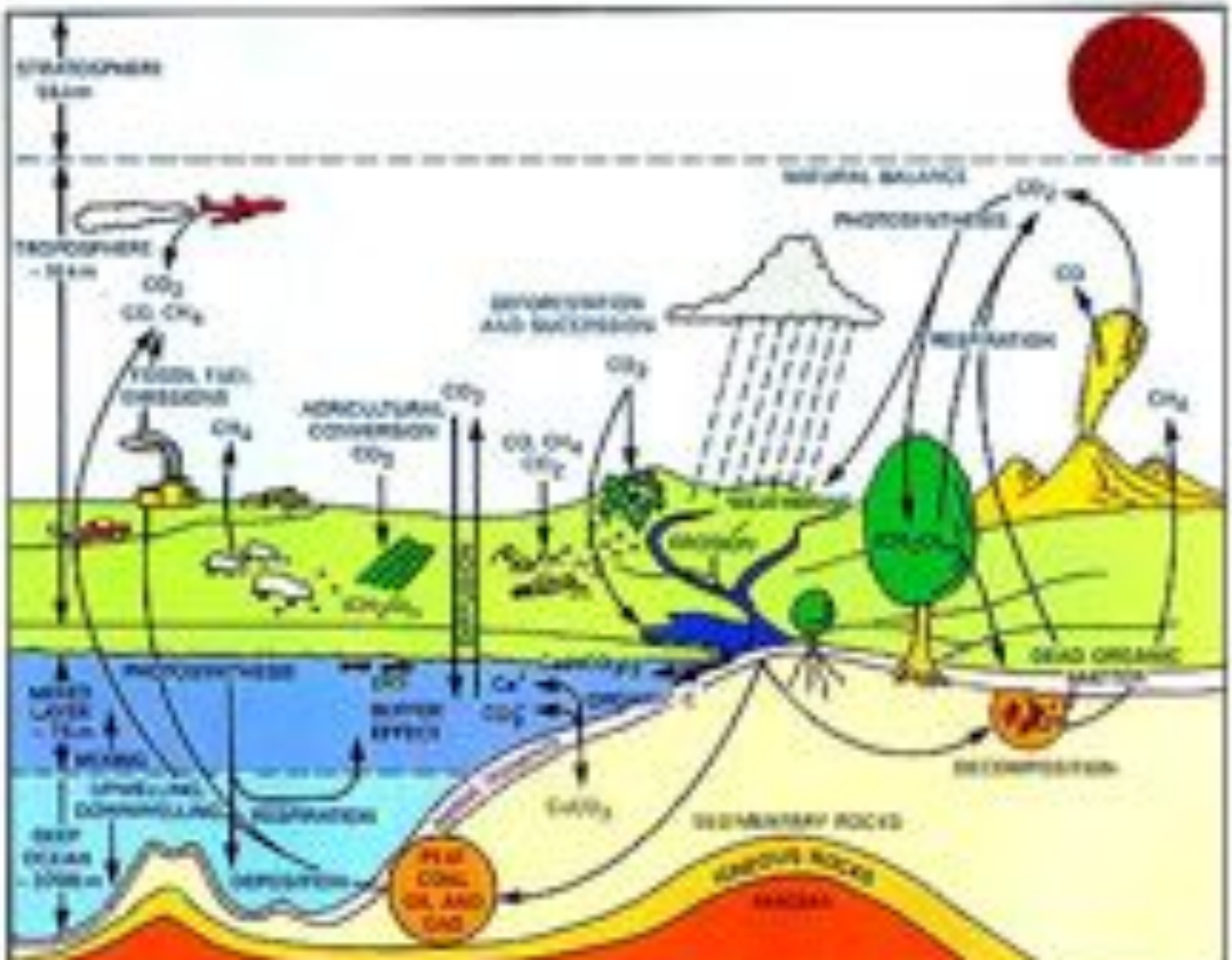
Making Security Measurable™

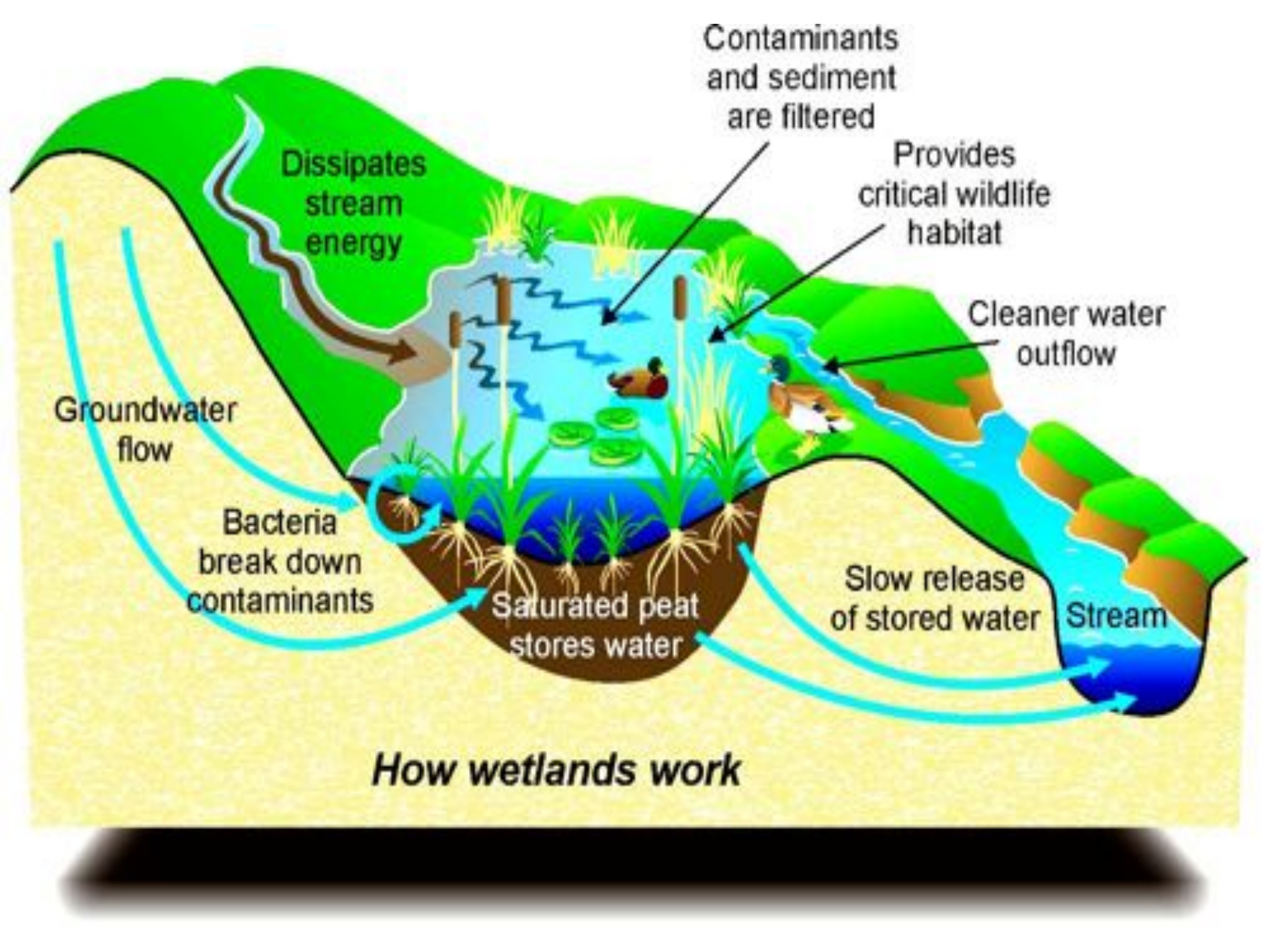
Global Oxygen Cycling on Earth

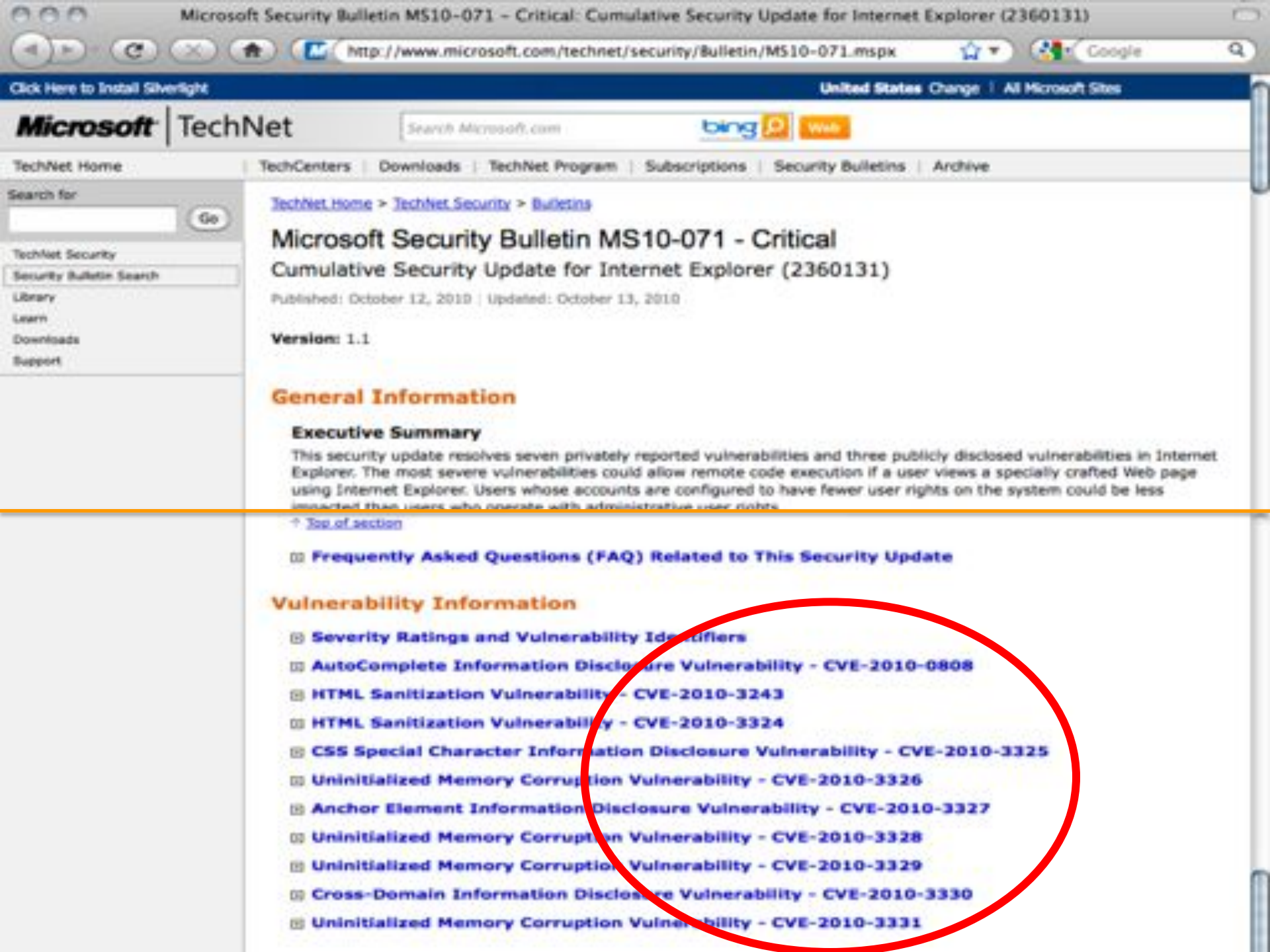




Making Security Measurable

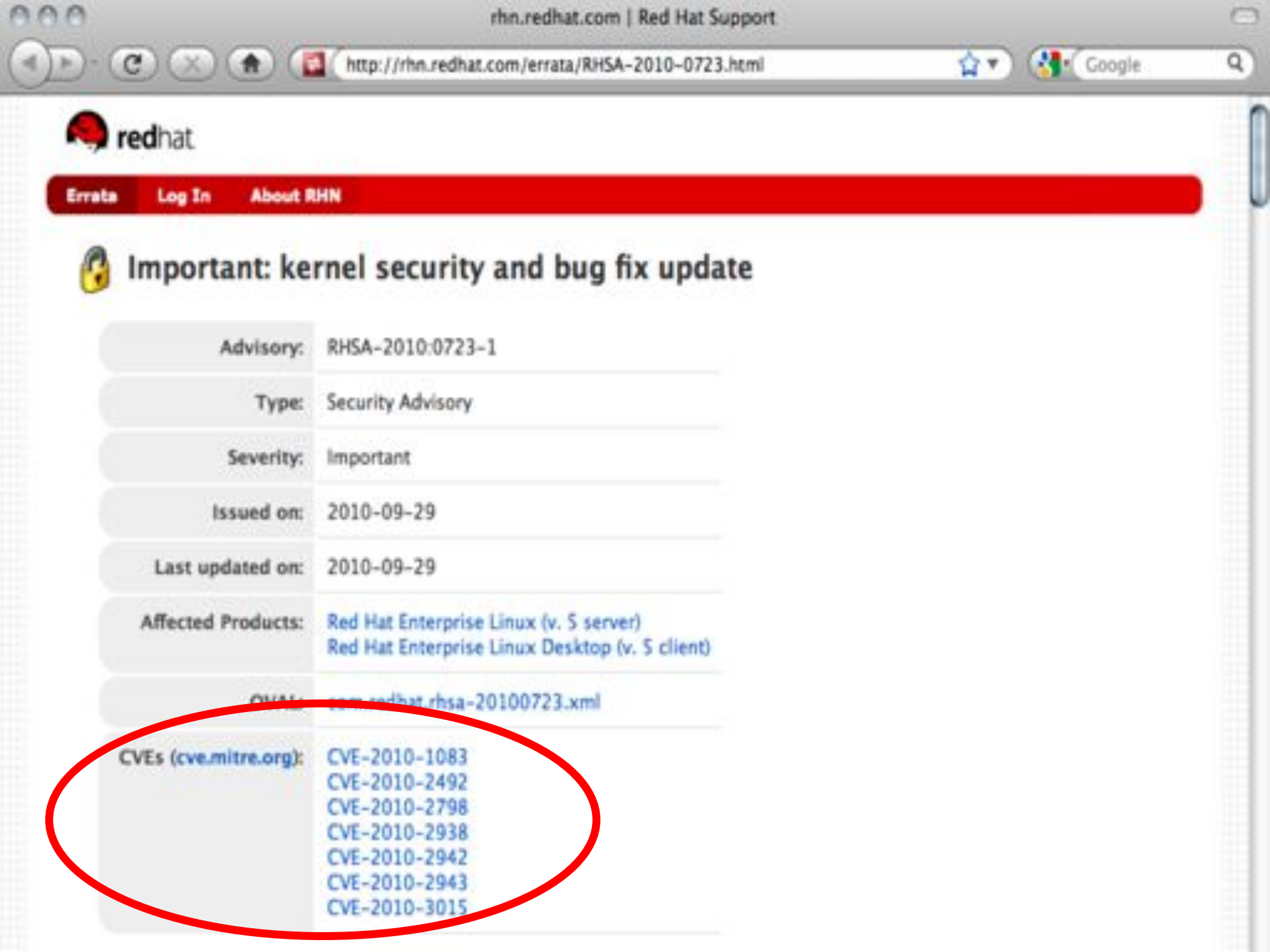

STRATOSPHERE
TROPOSPHERE
NATURAL BALANCE
PHOTOSYNTHESIS
RESPIRATION
DEFORESTATION AND SUCCESSION
FOSSIL FUEL EMISSIONS
AGRICULTURAL CONVERSION
MIXED LAYER
BUFFER EFFECT
DEEP OCEAN
DISSOLUTION
SEDIMENTARY ROCKS
PLANT DETRITUS
DEAD ORGANIC MATTER
DECOMPOSITION


Contaminants and sediment are filtered
Dissipates stream energy
Provides critical wildlife habitat
Cleaner water outflow
Groundwater flow
Bacteria break down contaminants
Saturated peat stores water
Slow release of stored water
Stream
How wetlands work


Microsoft Security Bulletin MS10-071 - Critical: Cumulative Security Update for Internet Explorer (2360131)
http://www.microsoft.com/technet/security/Bulletin/MS10-071.mspx
Click Here to Install Silverlight
United States Change | All Microsoft Sites
Microsoft TechNet
Search Microsoft.com
bing
Web
TechNet Home
TechCenters | Downloads | TechNet Program | Subscriptions | Security Bulletins | Archive
Search for
Go
TechNet Security
Security Bulletin Search
Library
Learn
Downloads
Support
TechNet Home > TechNet Security > Bulletins
Microsoft Security Bulletin MS10-071 - Critical
Cumulative Security Update for Internet Explorer (2360131)
Published: October 12, 2010 | Updated: October 13, 2010
Version: 1.1
General Information
Executive Summary
This security update resolves seven privately reported vulnerabilities and three publicly disclosed vulnerabilities in Internet Explorer. The most severe vulnerabilities could allow remote code execution if a user views a specially crafted Web page using Internet Explorer. Users whose accounts are configured to have fewer user rights on the system could be less
Top of section
Frequently Asked Questions (FAQ) Related to This Security Update
Vulnerability Information
Severity Ratings and Vulnerability Identifiers
AutoComplete Information Disclosure Vulnerability - CVE-2010-0808
HTML Sanitization Vulnerability - CVE-2010-3243
HTML Sanitization Vulnerability - CVE-2010-3324
CSS Special Character Information Disclosure Vulnerability - CVE-2010-3325
Uninitialized Memory Corruption Vulnerability - CVE-2010-3326
Anchor Element Information Disclosure Vulnerability - CVE-2010-3327
Uninitialized Memory Corruption Vulnerability - CVE-2010-3328
Uninitialized Memory Corruption Vulnerability - CVE-2010-3329
Cross-Domain Information Disclosure Vulnerability - CVE-2010-3330
Uninitialized Memory Corruption Vulnerability - CVE-2010-3331

Oracle Database Server Risk Matrix

CVE#	Component	Protocol	Package and/or Privilege Required	Remote Exploit without Auth.?	CVSS VERSION 2.0 RISK (see Risk Matrix Definitions)							Last Affected Patch set (per Supported Release)	Notes
					Base Score	Access Vector	Access Complexity	Authen-tication	Confiden-tiality	Integrity	Avail-ability		
CVE-2010-2390 (Oracle Enterprise Manager Grid Control)	EM Console	HTTP	None	Yes	7.5	Network	Low	None	Partial+	Partial+	Partial+	10.1.0.5, 10.2.0.3	See Note 1
CVE-2010-2419	Java Virtual Machine	Oracle Net	Create Session	No	6.5	Network	Low	Single	Partial+	Partial+	Partial+	10.1.0.5, 10.2.0.4, 11.1.0.7, 11.2.0.1	
CVE-2010-1321	Change Data Capture	Oracle Net	Execute on DBMS_CDC_PUBLISH	No	5.5	Network	Low	Single	Partial+	Partial+	None	-	See Note 2
CVE-2010-2412	OLAP	Oracle Net	Create Session	No	5.5	Network	Low	Single	Partial+	Partial+	None	11.1.0.7	
CVE-2010-2415	Change Data Capture	Oracle Net	Execute on DBMS_CDC_PUBLISH	No	4.9	Network	Medium	Single	Partial+	Partial+	None	10.1.0.5, 10.2.0.4, 11.1.0.7, 11.2.0.1	
CVE-2010-2411	Job Queue	Oracle Net	Execute on SYS.DBMS_IJOB	No	4.6	Network	High	Single	Partial+	Partial+	Partial+	-	See Note 2
CVE-2010-2407	XDK	HTTP	None	Yes	4.3	Network	Medium	None	None	Partial	None	10.1.0.5, 10.2.0.4, 11.1.0.7	
CVE-2010-2391	Core RDBMS	Oracle Net	Create Session	No	3.6	Network	High	Single	Partial	Partial	None	10.1.0.5, 10.2.0.3	
CVE-2010-2389 (Oracle Fusion Middleware)	Perl	Oracle Net	Local Logon	No	1.0	Local	High	Single	None	Partial+	None	-	See Note 2


rhn.redhat.com | Red Hat Support
http://rhn.redhat.com/errata/RHSA-2010-0723.html
Google
redhat
Errata
Log In
About RHN
Important: kernel security and bug fix update
Advisory: RHSA-2010:0723-1
Type: Security Advisory
Severity: Important
Issued on: 2010-09-29
Last updated on: 2010-09-29
Affected Products: Red Hat Enterprise Linux (v. 5 server)
Red Hat Enterprise Linux Desktop (v. 5 client)
rhsa-20100723.xml
CVEs (cve.mitre.org): CVE-2010-1083
CVE-2010-2492
CVE-2010-2798
CVE-2010-2938
CVE-2010-2942
CVE-2010-2943
CVE-2010-3015

APPLE-SA-2010-08-11-1 iOS 4.0.2 Update for iPhone and iPod touch

Subject: APPLE-SA-2010-08-11-1 iOS 4.0.2 Update for iPhone and iPod touch

From: Apple Product Security <email@hidden>

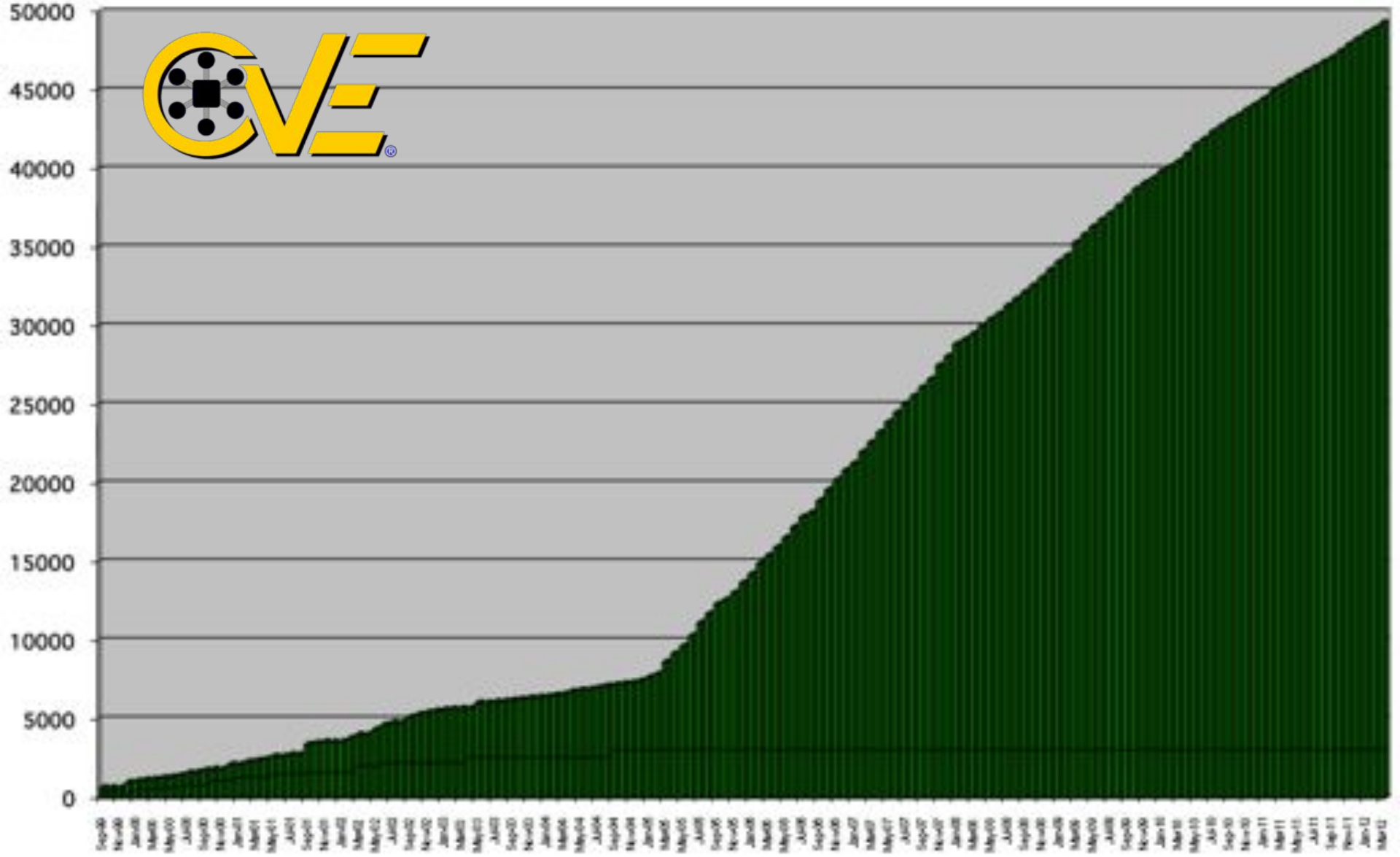
Date: Wed, 11 Aug 2010 12:19:43 -0700

Delivered-to: email@hidden

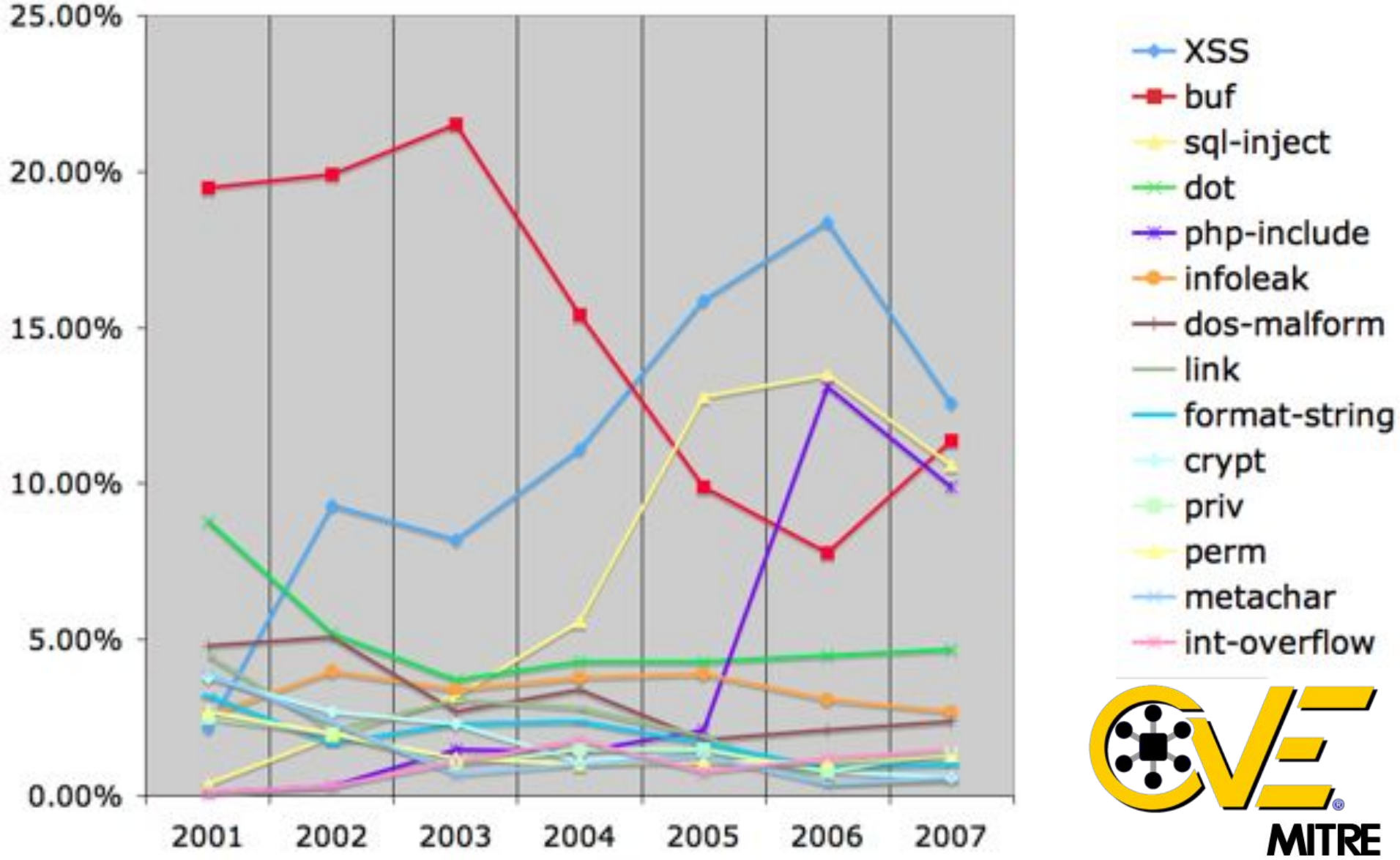
Delivered-to: email@hidden

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA1

APPLE-SA-2010-08-11-1 iOS 4.0.2 Update for iPhone and iPod touch

iOS 4.0.2 Update for iPhone and iPod touch is now available and
addresses the following:

FreeType
CVE-ID:  CVE-2010-1797
Available for:  iOS 2.0 through 4.0.1 for iPhone 3G and later,
iOS 2.1 through 4.0 for iPod touch (2nd generation) and later
Impact:  Viewing a PDF document with maliciously crafted embedded
fonts may allow arbitrary code execution
Description:  A stack buffer overflow exists in FreeType's handling
of CFF opcodes. Viewing a PDF document with maliciously crafted
```

CVE 1999 to 2012

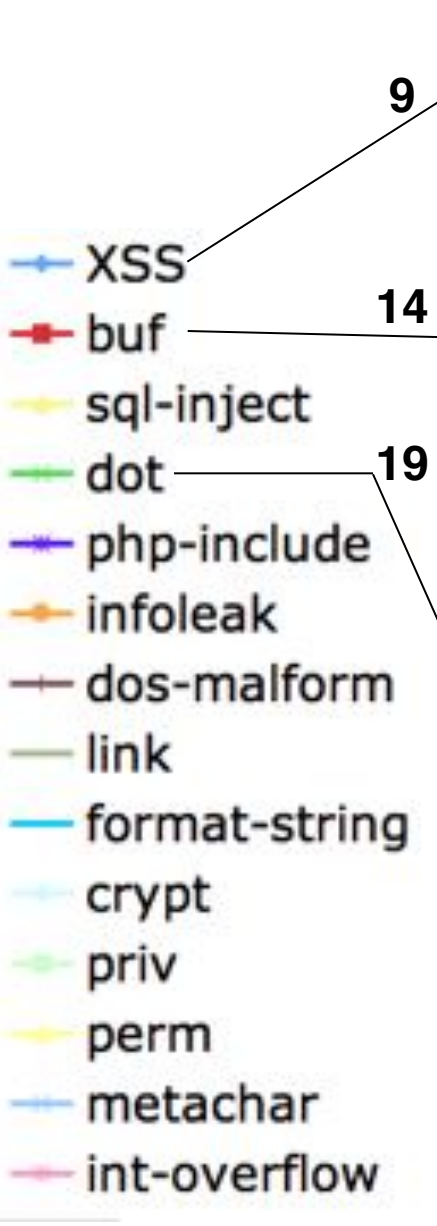


Vulnerability Type Trends: A Look at the CVE List (2001 - 2007)



CVE®
MITRE

Removing and Preventing the Vulnerabilities Requires More Specific Definitions…CWEs



9

Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') (79)
- Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS) (80)
- Improper Neutralization of Script in an Error Message Web Page (81)
- Improper Neutralization of Script in Attributes of IMG Tags in a Web Page (82)
- Improper Neutralization of Script in Attributes in a Web Page (83)
- Improper Neutralization of Encoded URI Schemes in a Web Page (84)
- Doubled Character XSS Manipulations (85)
- Improper Neutralization of Invalid Characters in Identifiers in Web Pages (86)
- Improper Neutralization of Alternate XSS Syntax (87)

14

Improper Restriction of Operations within the Bounds of a Memory Buffer (119)
- Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') (120)
- Write-what-where Condition (123)
- Out-of-bounds Read (125)
- Improper Handling of Length Parameter Inconsistency (130)
- Improper Validation of Array Index (129)
- Return of Pointer Value Outside of Expected Range (466)
- Access of Memory Location Before Start of Buffer (786)
- Access of Memory Location After End of Buffer (788)
- Buffer Access with Incorrect Length Value 805
- Untrusted Pointer Dereference (822)
- Use of Out-of-range Pointer Offset (823)
- Access of Uninitialized Pointer (824)
- Expired Pointer Dereference (825)

19

Path Traversal (22)
- Relative Path Traversal (23)
  - Path Traversal: '../filedir' (24)
  - Path Traversal: '/../filedir' (25)
  - <----------8 more here ---------->
  - Path Traversal: '..../' (34)
  - Path Traversal: '.../.../' (35)
- Absolute Path Traversal (36)
  - Path Traversal: '/absolute/pathname/here' (37)
  - Path Traversal: '\absolute\pathname\here' (38)
  - Path Traversal: 'C:dirname' (39)
  - Path Traversal: '\\UNC\share\name\' (Windows UNC Share) (40)

© 2012 MITRE

Weakness
Primary
Name
MITRE
Description
Concepts
Content
updated
CWE
Research
Internal
Team
Development
BaseWeakness
Date
Organization
Relationships
ParentOf
Taxonomy
Example
Notes
ClassWeakness
Mapped
ChildOf
Window
code
data
Definition
input
Time
Improper
Base
Source
Entry
ID
New
Mappings
Mitigations
Potential
Secure
View
Node
Variant
relationship
Previous
Weaknesses


Wouldn’t it be nice
if the weaknesses
in software were as
easy to spot and
their impact as
easy to understand
as a screen door in
a submarine…

a time-of-check-time-of-use (TOCTOU) bug that led to code calling into a freed memory block. The
on Weakness Enumeration (CWE) classification for this vulnerability is CWE-367.

Common Weakness Enumeration
A Community-Developed Dictionary of Software Weakness Types

CWE and SANS Institute TOP 25 MOST DANGEROUS SOFTWARE ERRORS

CWE-367: Time-of-check Time-of-use (TOCTOU) Race Condition

Time-of-check Time-of-use (TOCTOU) Race Condition

Weakness ID: 367 (Weakness Base) **Status:** Incomplete

Description

Description Summary

The software checks the state of a resource before using that resource, but the resource's state can change between the check and the use in a way that invalidates the results of the check. This can cause the software to perform invalid actions when the resource is in an unexpected state.

Extended Description

This weakness can be security-relevant when an attacker can influence the state of the resource between check and use. This can happen with shared resources such as files, memory, or even variables in multithreaded programs.

Alternate Terms

TOCTTOU: The TOCCTOU acronym expands to "Time Of Check To Time Of Use". Usage varies between TOCTOU and TOCTTOU.

Time of Introduction

- Implementation

Applicable Platforms

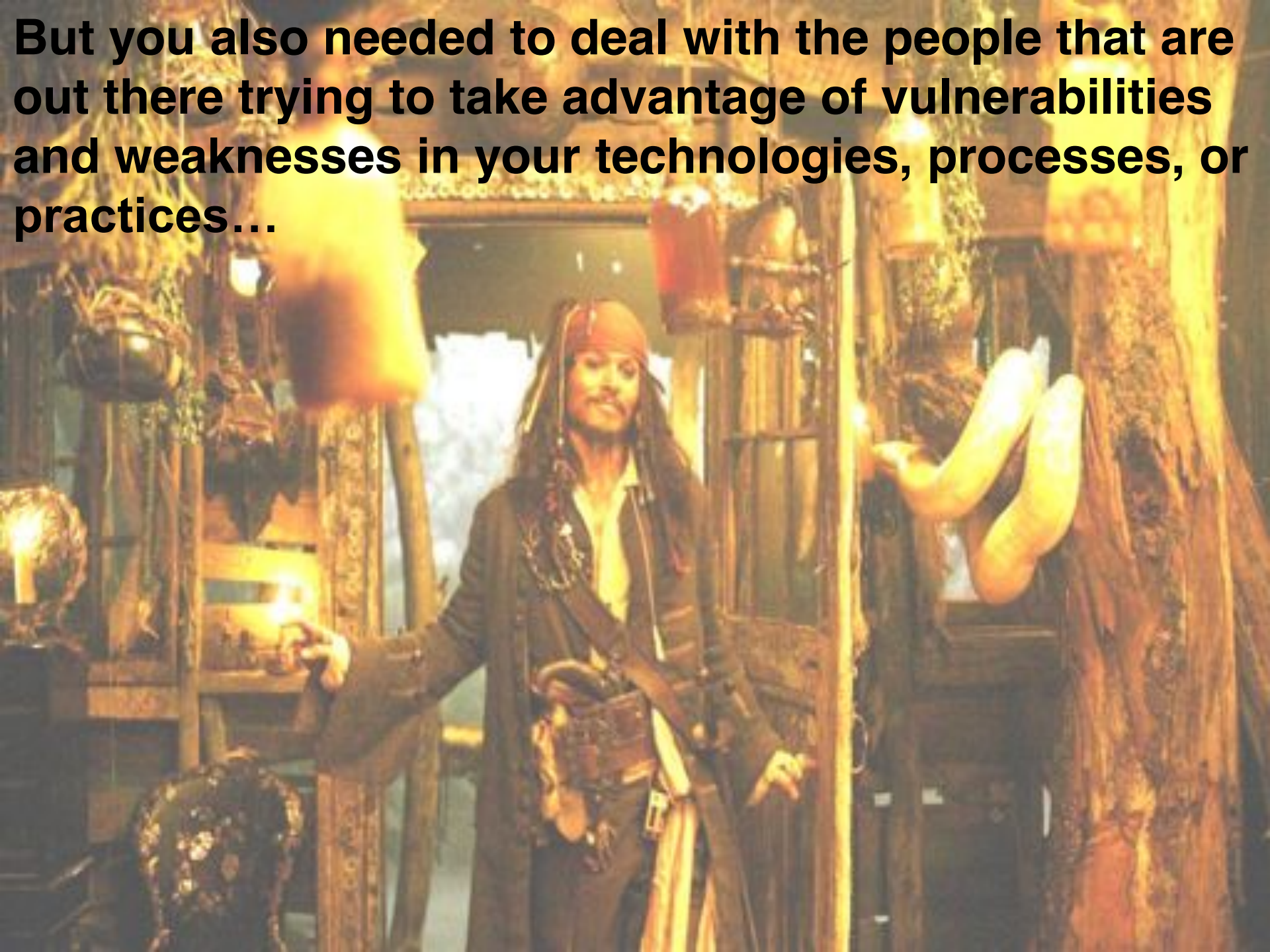
Languages

All

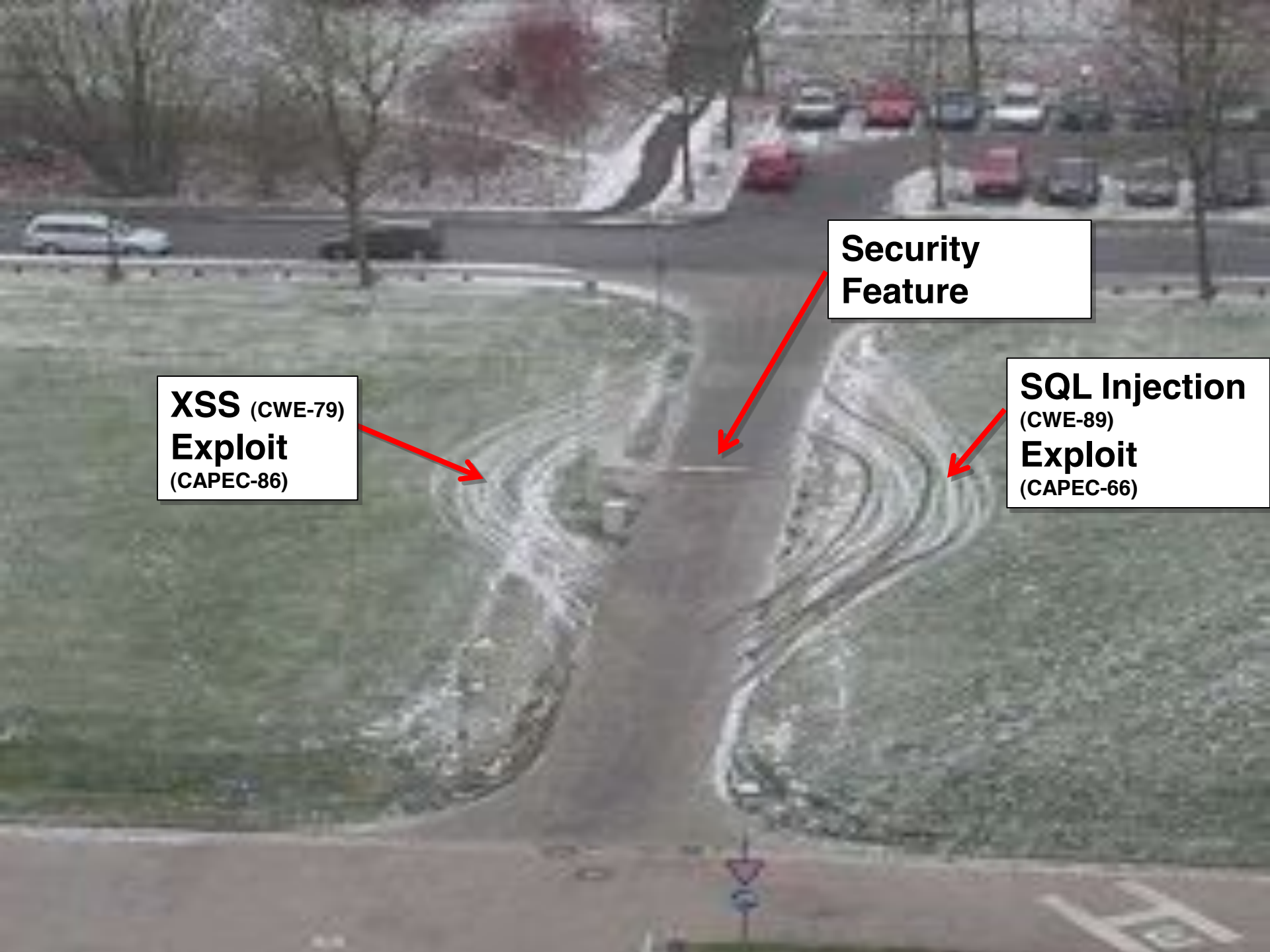
Common Consequences

Scope	Effect
Access Control	The attacker can gain access to otherwise unauthorized resources.
Access Control Authorization	Race conditions such as this kind may be employed to gain read or write access to resources which are not normally readable or writable by the user in question.
Integrity	The resource in question, or other resources (through the corrupted one), may be changed in undesirable ways by a malicious user.
Accountability	If a file or other resource is written in this method, as opposed to in a valid way, logging of the activity may not occur.
Non-Repudiation	In some cases it may be possible to delete files a malicious user might not otherwise have access to, such as log files.

But you also needed to deal with the people that are out there trying to take advantage of vulnerabilities and weaknesses in your technologies, processes, or practices…




…with defensive and offensive security capabilities.
712


Security Feature
XSS (CWE-79) Exploit (CAPEC-86)
SQL Injection (CWE-89) Exploit (CAPEC-66)

SQL Injection Attack Execution Flow

MS SQL Database

1. Web Form with ‘ in all fields

2. One SQL error message

```
SELECT ITEM,PRICE FROM
PRODUCT WHERE
ITEM_CATEGORY='$user_input'
ORDER BY PRICE
```

5. Web Form with: ' exec master..xp_cmdshell 'dir' --

6. a listing of all directories

© 2012 MITRE

Simple test case for SQL Injection

Test Case 1: Single quote SQL injection of registration page web form fields

Test Case Goal: Ensure SQL syntax single quote character entered in registration page web form fields does not cause abnormal SQL behavior

Context:

- This test case is part of a broader SQL injection syntax exploration suite of tests to probe various potential injection points for susceptibility to SQL injection. If this test case fails, it should be followed-up with test cases from the SQL injection experimentation test suite.

Preconditions:

- Access to system registration page exists
- Registration page web form field content are used by system in SQL queries of the system database upon page submission
- User has the ability to enter free-form text into registration page web form fields

Test Data:

- ASCII single quote character

Action Steps:

- Enter single quote character into each web form field on the registration page
- Submit the contents of the registration page

Postconditions:

- Test case fails if SQL error is thrown
- Test case passes if page submission succeeds without any SQL errors

© 2012 MITRE

Common Weakness Enumeration
A Community-Developed Dictionary of Software Weakness Types

TOP 25 MOST DANGEROUS SOFTWARE ERRORS

CWE-89: Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')

Weakness ID: 89 (Weakness Base) **Status:** Draft

Description

Description Summary

The software constructs all or part of an SQL command using externally-influenced input from an upstream component, but it does not neutralize or incorrectly neutralizes special elements that could modify the intended SQL command when it is sent to a downstream component.

Extended Description

Without sufficient removal or quoting of SQL syntax in user-controllable inputs, the generated SQL query can cause those inputs to be interpreted as SQL instead of ordinary user data. This can be used to alter query logic to bypass security checks, or to insert additional statements that modify the back-end database, possibly including execution of system commands.

SQL injection has become a common issue with database-driven web sites. The flaw is easily detected, and easily exploited, and as such, any site or software package with even a minimal user base is likely to be subject to an attempted attack of this kind. This flaw depends on the fact that SQL makes no real distinction between the control and data planes.

Time of Introduction

- Architecture and Design
- Implementation
- Operation

Applicable Platforms

Languages

All

Technology Classes

Database-Server

MITRE

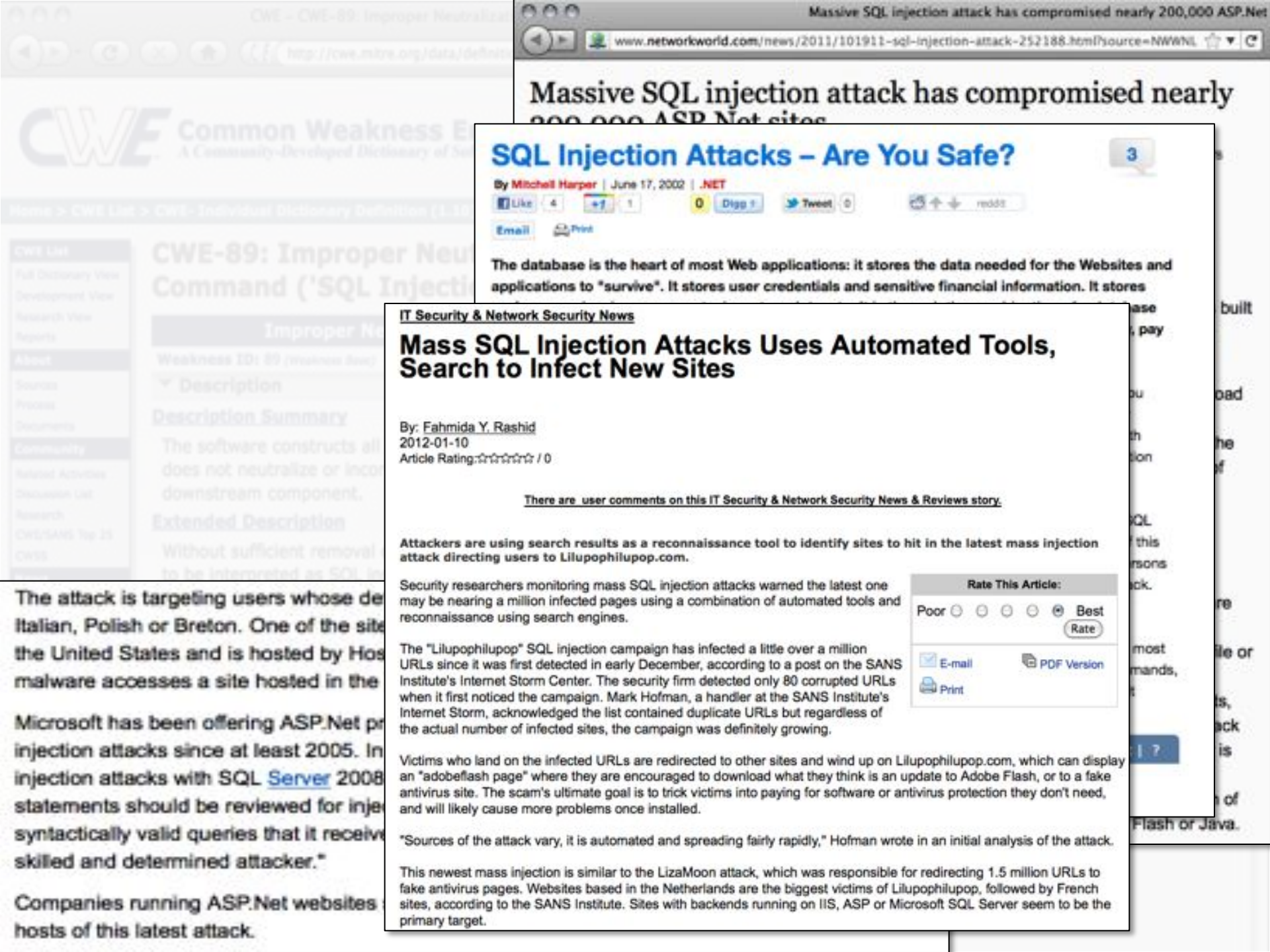
SANS

Rank	Score	ID	Name		
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')	up from 2	+1
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')	up from 9	+7
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')	same	0
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')	down from 1	-3
[5]	76.9	CWE-306	Missing Authentication for Critical Function	up from 19	+14
[6]	76.8	CWE-862	Missing Authorization	split of prior #5	-1
[7]	75.0	CWE-798	Use of Hard-coded Credentials	up from 11	+4
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data	up from 10	+2
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type	down from 8	-1
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision	down from 6	-4
[11]	73.1	CWE-250	Execution with Unnecessary Privileges	new entry	n/a
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)	down from 4	-8
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')	down from 7	-6
[14]	68.5	CWE-494	Download of Code Without Integrity Check	up from 20	+6
[15]	67.8	CWE-863	Incorrect Authorization	split of prior #5	-10

TOP 25

©


Google Earth
Severity
Emergency
Critical
Warning
SQL Injection Probe Detected
Image IBCAO
Image © 2010 TerraMetrics
Image USDA Farm Service Agency
Google



Massive SQL injection attack has compromised nearly 200,000 ASP.Net

www.networkworld.com/news/2011/101911-sql-injection-attack-252188.html?source=NWWNL

Massive SQL injection attack has compromised nearly

SQL Injection Attacks – Are You Safe?

3

By Mitchell Harper | June 17, 2002 | .NET

Like 4 +1 1 0 Digg ! Tweet 0 reddit

Email Print

The database is the heart of most Web applications: it stores the data needed for the Websites and applications to "survive". It stores user credentials and sensitive financial information. It stores

IT Security & Network Security News

Mass SQL Injection Attacks Uses Automated Tools, Search to Infect New Sites

By: Fahmida Y. Rashid
2012-01-10
Article Rating:☆☆☆☆☆ / 0

There are user comments on this IT Security & Network Security News & Reviews story.

Attackers are using search results as a reconnaissance tool to identify sites to hit in the latest mass injection attack directing users to Lilupophilupop.com.

Rate This Article:
Poor ○ ○ ○ ○ ○ Best
Rate

E-mail PDF Version
Print

Security researchers monitoring mass SQL injection attacks warned the latest one may be nearing a million infected pages using a combination of automated tools and reconnaissance using search engines.

The "Lilupophilupop" SQL injection campaign has infected a little over a million URLs since it was first detected in early December, according to a post on the SANS Institute's Internet Storm Center. The security firm detected only 80 corrupted URLs when it first noticed the campaign. Mark Hofman, a handler at the SANS Institute's Internet Storm, acknowledged the list contained duplicate URLs but regardless of the actual number of infected sites, the campaign was definitely growing.

Victims who land on the infected URLs are redirected to other sites and wind up on Lilupophilupop.com, which can display an "adobeflash page" where they are encouraged to download what they think is an update to Adobe Flash, or to a fake antivirus site. The scam's ultimate goal is to trick victims into paying for software or antivirus protection they don't need, and will likely cause more problems once installed.

"Sources of the attack vary, it is automated and spreading fairly rapidly," Hofman wrote in an initial analysis of the attack.

This newest mass injection is similar to the LizaMoon attack, which was responsible for redirecting 1.5 million URLs to fake antivirus pages. Websites based in the Netherlands are the biggest victims of Lilupophilupop, followed by French sites, according to the SANS Institute. Sites with backends running on IIS, ASP or Microsoft SQL Server seem to be the primary target.

The attack is targeting users whose de
Italian, Polish or Breton. One of the site
the United States and is hosted by Hos
malware accesses a site hosted in the

Microsoft has been offering ASP.Net pr
injection attacks since at least 2005. In
injection attacks with SQL Server 2008
statements should be reviewed for inje
syntactically valid queries that it receiv
skilled and determined attacker."

Companies running ASP.Net websites
hosts of this latest attack.

CWE web site visitors by City

IBM Software
Technical White Paper

One way to improve software security is to gain a better understanding of the most common weaknesses that can affect software security. With that in mind, there are many resources available online to help organizations learn about

Resources available to help organizations protect systems in

Resource	Focus
DoD Information Assurance Certification and Accreditation Process (DIACAP)	The DIACAP defines the minimum standa... accredited by the DoD and authorized to... application-level security controls, but it i... activities, general tasks, and a managem...
Defense Information Systems Agency (DISA)	The DISA provides a security technical i... development that offer more granular inf... bility assessment techniques. The checklist is the same one used by DoD auditors.
U.S. Department of Homeland Security (DHS)	The DHS offers information on security best practices and tools for application- and soft... part of its "Build Security In" initiative.
The Common Weaknesses Enumeration project, a community-based program sponsored by the MITRE Corporation, an IBM Business Partner	The MITRE Corporation maintains the online common vulnerabilities and exposures (CV... enumeration (CWE) knowledge bases about currently known vulnerabilities and types of... knowledge base focuses on packaged software and deals with patches and known vul... knowledge base focuses on code vulnerabilities.
The Open Web Application Security Project (OWASP)	One of the best sources for information on web application security issues, the OWASP... 10 list of the most dangerous and most commonly found and commonly exploited vulne... how to identify, fix and avoid them.
Cigital Building Security In Maturity Model (BSIMM)	Created by Cigital, an IBM Business Partner, the BSIMM is designed to help organizatio... and plan a software security initiative. The focus is on making applications more secure,... process and at later stages in the software life cycle.
IBM X-Force™ research and development team	A global cyberthreat and risk analysis team that monitors traffic and attacks around the ... IBM X-Force team is an excellent resource for trend analysis and answers to questions a... attacks are most common, where they are coming from and what organizations can do ... the risks.
IBM Institute for Advanced Security (IAS)	This companywide cybersecurity initiative applies IBM research, services, software and t... help governments and other clients improve the security and resiliency of their IT and bu...

Test and vulnerability assessment

Testing applications for security defects should be an integral and organic part of any software testing process. During security testing, organizations should test to help ensure that the security requirements have been implemented and the product is free of vulnerabilities.

The SEF refers to the MITRE Common Weakness Enumeration[5] (CWE) list and the Common Vulnerability E... be tested. Thi... information a... and vulnerabi... against the m...

Creating a se... plan includes ...

[5] For more inform...
[6] For more inform...

IBM

Security in Development: The IBM Secure Engineering Framework

Redguides
for Business Leaders

Danny Allan
Tim Hahn
Andras Szakal
Jim Whitmore
Axel Buecker

- Investigating common development processes and the IBM Integrated Product Development process
- Emphasizing security awareness and requirements in the software development process
- Discussing test and vulnerability assessments

Redbooks

Software Engineering Institute

Making the Business Case for Software Assurance

Nancy R. Mead
Julia H. Allen
W. Arthur Conklin
Antonio Drommi
John Harrison
Jeff Ingalsbe
James Rainey
Dan Shoemaker

April 2009

SPECIAL REPORT
CMU/SEI-2009-SR-001

CERT Program
Unlimited distribution subject to the copyright.

http://www.sei.cmu.edu

Carnegie Mellon

OVM: An Ontology for Vulnerability Management

Ju An Wang & Minzhe Guo
Southern Polytechnic State University
1100 South Marietta Parkway
Marietta, GA 30060
(01) 678-915-3718

jwang@spsu.edu

ABSTRACT

In order to reach the goals of the Information Security Automation Program (ISAP) [1], we propose an ontological approach to capturing and utilizing the fundamental concepts in information security and their relationship, retrieving vulnerability data and reasoning about the cause and impact of vulnerabilities. Our ontology for vulnerability management (OVM) has been populated with all vulnerabilities in NVD [2] with additional inference rules, knowledge representation, and data-mining mechanisms. With the seamless integration of common vulnerabilities and their related concepts such as attacks and countermeasures, OVM provides a promising pathway to making ISAP successful.

Categories and Subject Descriptors

C.2.0 [**Computer-Communication Networks**]: General [Security and protection]; K.6.5 [**Management of Computing and Information Systems**]: Security and Protection;

General Terms

Ontology, Security, Vulnerability Analysis and Management

Keywords

Security vulnerability, Semantic technology, Ontology, Vulnerability analysis

1. INTRODUCTION

The Information Security Automation Program (ISAP) is a U.S. government multi-agency initiative to enable automation and standardization of technical security operations [1]. Its high-level goals include standards based automation of security checking and remediation as well as automation of technical compliance activities. Its low-level objectives include enabling standards based communication of vulnerability data, customizing and managing configuration baselines for various IT products, assessing information systems and reporting compliance status, using standard metrics to weight and aggregate potential vulnerability impact, and remediating identified vulnerabilities [1]. Secure computer systems ensure that confidentiality, integrity, and availability are maintained for users, data, and other information assets. Over the past a few decades, a significantly large amount of knowledge has been accumulated in the area of information security. However, a lot of concepts in information security are vaguely defined and sometimes they have different semantics in different contexts, causing misunderstanding among stake holders due to the language ambiguity. On the other hand, the standardization, design and development of security tools [1-3] require a systematic classification and definition of security concepts and techniques. It is important to have a clearly defined vocabulary and standardized language as means to accurately communicate system vulnerability information and their countermeasures among all the people involved. We believe that semantic technology in general, and ontology in particular, could be a useful tool for system security. Our research work has confirmed this belief and this paper will report some of our work in this area.

An ontology is a specification of concepts and their relationship. Ontology represents knowledge in a formal and structured form. Therefore, ontology provides a better tool for communication, reusability, and organization of knowledge. Ontology is a knowledge representation (KR) system based on Description Logics (DLs) [8], which is an umbrella name for a family of KR formalisms representing knowledge in various domains. The DL formalism specifies a knowledge domain as the "world" by first defining the relevant concepts of the domain, and then it uses these concepts to specify properties of objects and individuals occurring in the domain [10-12]. Semantic technologies not only provide a tool for communication, but also a foundation for high-level reasoning and decision-making. Ontology, in particular, provides the potential of formal logic inference based on well-defined data and knowledge bases. Ontology captures the relationships between collected data and uses the explicit knowledge of concepts and relationships to deduce the implicit and inherent knowledge. As a matter of fact, a heavy-weight ontology could be defined as a formal logic system, as it includes facts and rules, concepts, concept taxonomies, relationships, properties, axioms and constraints.

A vulnerability is a security flaw, which arises from computer system design, implementation, maintenance, and operation. Research in the area of vulnerability analysis focuses on discovery of previously unknown vulnerabilities and quantification of the security of systems according to some metrics. Researchers at MITRE have provided a standard format for naming a security vulnerability, called Common Vulnerabilities and Exposures (CVE) [14], which assigns each vulnerability a unique identification number. We have designed a vulnerability ontology (OVM (ontology for vulnerability management) populated with all existing vulnerabilities in NVD [2]. It supports research on reasoning about vulnerabilities and characterization of vulnerabilities and their impact on computing systems. Vendors and users can use our ontology in support of vulnerability analysis, tool development and vulnerability management.

The rest of this paper is organized as follows: Section 2 presents the architecture of our OVM. Section 3 discusses how to populate the OVM with vulnerability instances from NVD and other

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

CSIIRW '08, April 13-15, Oak Ridge, Tennessee, USA
Copyright © 2009 ACM 978-1-60558-518-5 ... $5.00

16 July 2010

A Human Capital Crisis in Cybersecurity

Technical Proficiency Matters

A White Paper of the
CSIS Commission on Cybersecurity for the 44th Presidency

COCHAIRS
Representative James R. Langevin
Representative Michael T. McCaul
Scott Charney
Lt. General Harry Raduege, USAF (ret.)

based on a body of knowledge that represents the complete set of concepts, terms and activities that make up a professional domain. And absent such a body of knowledge there is little basis for supporting a certification program. Indeed it would be dangerous and misleading.

A complete body of knowledge covering the entire field of software engineering may be years away. However, the body of knowledge needed by professionals to create software free of common and critical security flaws has been developed, vetted widely and kept up to date. That is the foundation for a certification program in software assurance that can gain wide adoption. It was created in late 2008 by a consortium of national experts, sponsored by DHS and NSA, and was updated in late 2009. It contains ranked lists of the most common errors, explanations of why the errors are dangerous, examples of those errors in multiple languages, and ways of eliminating those errors. It can be found at http://cwe.mitre.org/top25.

Any programmer who writes code without being aware of those problems and is not capable of writing code free of those errors is a threat to his or her employers and to others who use computers connected to systems running his or her software.

A complete body of knowledge covering the entire field of software engineering may be years away. However, the body of knowledge needed by professionals to create software free of common and critical security flaws has been developed, vetted widely and kept up to date. That is the foundation for a certification program in software assurance that can gain wide adoption. It was created in late 2008 by a consortium of national experts, sponsored by DHS and NSA, and was updated in late 2009. It contains ranked lists of the most common errors, explanations of why the errors are dangerous, examples of those errors in multiple languages, and ways of eliminating those errors. It can be found at http://cwe.mitre.org/top25.

Any programmer who writes code without being aware of those problems and is not capable of writing code free of those errors is a threat to his or her employers and to others who use computers connected to systems running his or her software.

© 2012 MITRE

CSSLP

The **Certified Secure Software Lifecycle Professional** (CSSLP) Certification Program will show software lifecycle stakeholders not only how to implement security, but how to glean security requirements, design, architect, test and deploy secure software.

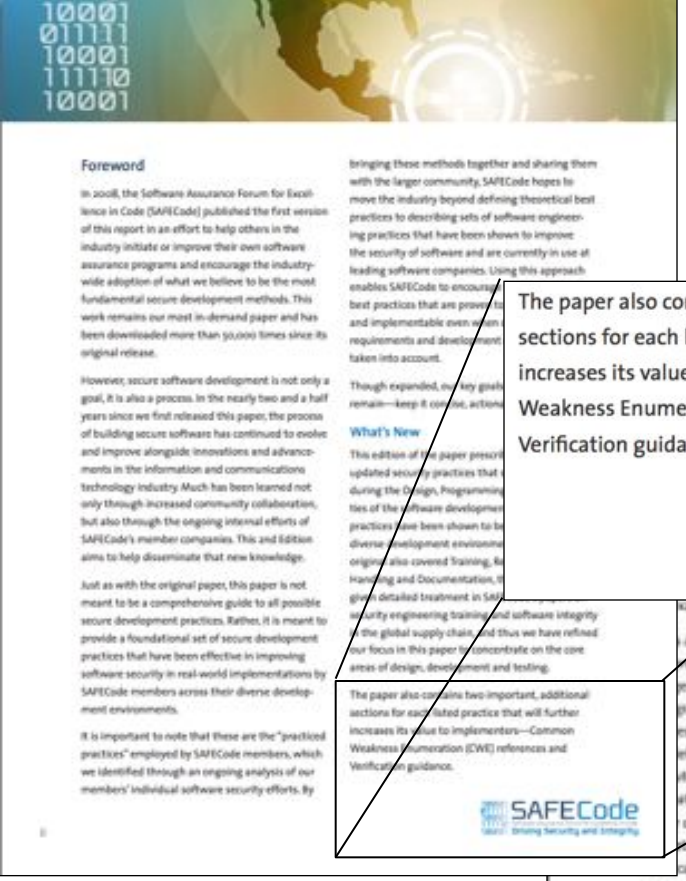
An Overview of the Steps:

(ISC)² ® 5-day CSSLP CBK® Education Program

Educate yourself and learn security best practices and industry standards for the software lifecycle through the CSSLP Education Program.(ISC)² provides education your way to fit your life and schedule.Completing this course will, not only teach all of the ...stablish a security plan across your

COMPUTER BASED TESTING NOW AVAILABLE FOR THE CSSLP

Industry Uptake



Foreword

In 2008, the Software Assurance Forum for Excellence in Code (SAFECode) published the first version of this report in an effort to help others in the industry initiate or improve their own software assurance programs and encourage the industry-wide adoption of what we believe to be the most fundamental secure development methods. This work remains our most in-demand paper and has been downloaded more than 50,000 times since its original release.

However, secure software development is not only a goal, it is also a process. In the nearly two and a half years since we first released this paper, the process of building secure software has continued to evolve and improve alongside innovations and advancements in the information and communications technology industry. Much has been learned not only through increased community collaboration, but also through the ongoing internal efforts of SAFECode's member companies. This 2nd Edition aims to help disseminate that new knowledge.

Just as with the original paper, this paper is not meant to be a comprehensive guide to all possible secure development practices. Rather, it is meant to provide a foundational set of secure development practices that have been effective in improving software security in real-world implementations by SAFECode members across their diverse development environments.

It is important to note that these are the "practiced practices" employed by SAFECode members, which we identified through an ongoing analysis of our members' individual software security efforts. By bringing these methods together and sharing them with the larger community, SAFECode hopes to move the industry beyond defining theoretical best practices to describing sets of software engineering practices that have been shown to improve the security of software and are currently in use at leading software companies. Using this approach enables SAFECode to encourage best practices that are proven and implementable even when requirements and development are taken into account.

Though expanded, our key goals remain—keep it concise, actionable

What's New

This edition of the paper prescribes updated security practices that during the Design, Programming ties of the software development practices have been shown to be diverse development environments original, also covered Training, Handling and Documentation, gives detailed treatment in SA security engineering training and software integrity in the global supply chain, and thus we have refined our focus in this paper to concentrate on the core areas of design, development and testing.

The paper also contains two important, additional sections for each listed practice that will further increases its value to implementers—Common Weakness Enumeration (CWE) references and Verification guidance.

SAFECode

The paper also contains two important, additional sections for each listed practice that will further increases its value to implementers—Common Weakness Enumeration (CWE) references and Verification guidance.

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

SAFECode
Driving Security and Integrity

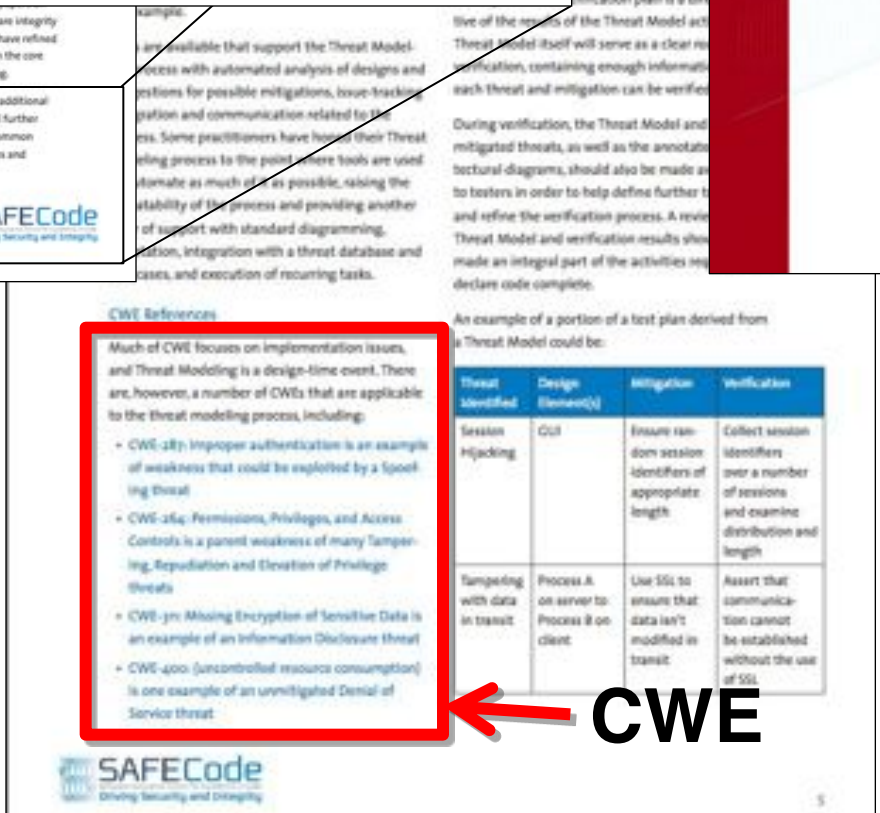
Fundamental Practices for Secure Software Development
2ND EDITION

A Guide to the Most Effective Secure Development Practices in Use Today

February 8, 2011

Editor Stacy Simpson, SAFECode

Authors
Mark Belk, Juniper Networks
Matt Coles, EMC Corporation
Cassio Goldschmidt, Symantec Corp.
Michael Howard, Microsoft Corp.
Kyle Randolph, Adobe Systems Inc.
Mikko Saario, Nokia
Reeny Sondhi, EMC Corporation
Izar Tarandach, EMC Corporation
Antti Vähä-Sipilä, Nokia
Yonko Yonchev, SAP AG



CWE References

Much of CWE focuses on implementation issues, and Threat Modeling is a design-time event. There are, however, a number of CWEs that are applicable to the threat modeling process, including:

- CWE-287: Improper authentication is an example of weakness that could be exploited by a Spoofing threat
- CWE-264: Permissions, Privileges, and Access Controls is a parent weakness of many Tampering, Repudiation and Elevation of Privilege threats
- CWE-311: Missing Encryption of Sensitive Data is an example of an Information Disclosure threat
- CWE-400: (uncontrolled resource consumption) is one example of an unmitigated Denial of Service threat

During verification, the Threat Model and mitigated threats, as well as the annotated to testers in order to help define further and refine the verification process. A revi Threat Model and verification results sho made an integral part of the activities re declare code complete.

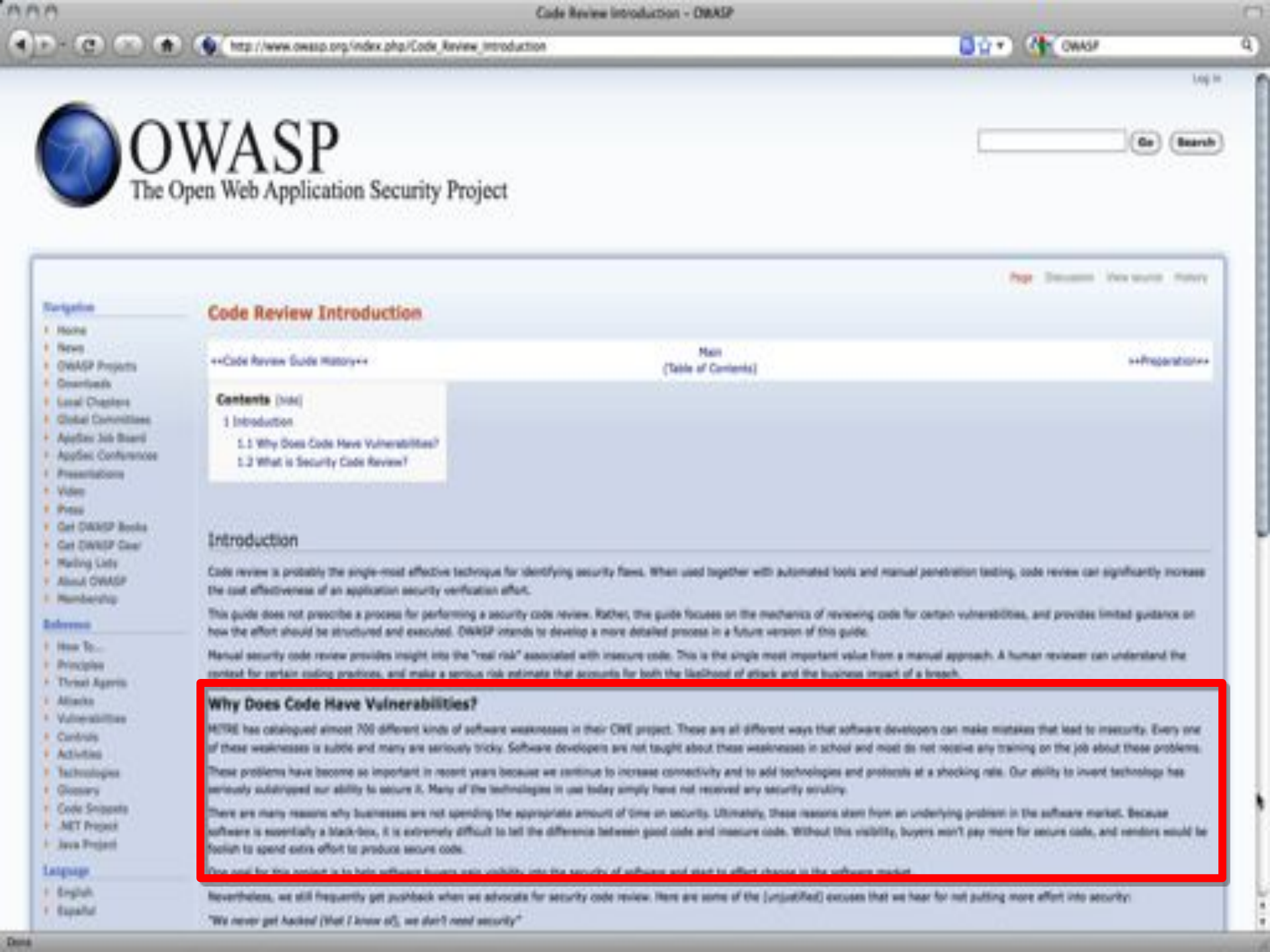
An example of a portion of a test plan derived from a Threat Model could be:

Threat Identified	Design Element(s)	Mitigation	Verification
Session Hijacking	GUI	Ensure random session identifiers of appropriate length	Collect session identifiers over a number of sessions and examine distribution and length
Tampering with data in transit	Process A on server to Process B on client	Use SSL to ensure that data isn't modified in transit	Assert that communication cannot be established without the use of SSL

SAFECode
Driving Security and Integrity

CWE

© 2012 MITRE


Code Review Introduction - OWASP
http://www.owasp.org/index.php/Code_Review_Introduction
OWASP
The Open Web Application Security Project
Code Review Introduction
Contents
1 Introduction
1.1 Why Does Code Have Vulnerabilities?
1.2 What is Security Code Review?
Introduction
Why Does Code Have Vulnerabilities?

Threat Classification Taxonomy Cross Reference View

last edited by Robert Auger 10 months, 3 weeks ago

Page history

Tags: Threat Classification

Check for plagiarism

Threat Classification 'Taxonomy Cross Reference View'

This view contains a mapping of the WASC Threat Classification's Attacks and Weaknesses with MITRE's Common Weakness Enumeration, MITRE's Common Attack Pattern Enumeration and Classification, OWASP Top Ten 2010 RC1 (original mapping with OWASP Top Ten from Jeremiah Grossman & Bill Corry) and SANS/CWE and OWASP Top Ten 2007 and 2004 (original mapping from Dan Cornell, Denim Group)

WASC ID	Name	CWE ID	CAPEC ID	SANS/CWE Top 25 2009	OWASP Top Ten 2010	OWASP Top Ten 2007	OWASP Top Ten 2004
WASC-01	Insufficient Authentication	287		642	A3 – Broken Authentication and Session Management, A4 – Insecure Direct Object References	A7 – Broken Authentication and Session Management, A4 – Insecure Direct Object Reference	A3 – Broken Authentication and Session management, A2 – Broken Access Control
WASC-02	Insufficient Authorization	284		285	A4 – Insecure Direct Object References, A7 – Failure to Restrict URL Access	A10 – Failure to Restrict URL Access, A4 – Insecure Direct Object Reference	A2 – Broken Access Control
WASC-03	Integer Overflows	190	128	682			
WASC-04	Insufficient Transport Layer Protection	311 523		319	A9 – Insufficient Transport Layer Protection	A9 – Insecure Communications	
WASC-05	Remote File Inclusion	98	193 253	426		A3 – Malicious File Execution	
WASC-06	Format String	134	67				
WASC-07	Buffer Overflow	119 120	10 100	119			A5 – Buffer Overflows
WASC-08	Cross-site Scripting	79	18 19 63	79	A2 – Cross-Site Scripting	A1 – Cross Site Scripting (XSS)	A4 – Cross Site Scripting (XSS)
WASC-09	Cross-site Request Forgery	352	62	352	A5 – Cross-Site Request Forgery	A5 – Cross Site Request Forgery (CSRF)	
WASC-10	Denial of Service	400	119	404	A10 – Failure to Restrict URL Access	A10 – Failure to Restrict URL Access	A9 – Denial of Service

Similar

WASC Projects
- Distributed Open Proxy Honeypots
- Script Mapping
- The Web Security Glossary
- Web Application Firewall Evaluation Criteria
- Web Application Security Scanner Evaluation Criteria
- Web Application Security Statistics
- Web Hacking Incidents Database
- WASC Threat Classification

WASC Project Leaders
- Robert Auger
- Ryan Barnett
- Romain Gaucher
- Sergey Gorbilnik
- Ofer Shezaf
- Brian Shura

WASC Main Website
- http://www.webappsec.org/

WASC Mailing Lists
- http://lists.webappsec.org/

WASC on Twitter
- http://twitter.com/wascupdates

Join us on LinkedIn!
- http://www.linkedin.com/groups?gid=83338

Recent Activity

Insufficient Data Protection Working

ISO/IEC JTC 1/SC 27/WG 3, NWP

Refining Software Vulnerability Analysis Under ISO/IEC 15408 and ISO/IEC 18045

- The way how the CAPEC and related CWE taxonomies are to be used by the developer, which needs to consider and provide sufficient and effective mitigation to all applicable attacks and weaknesses.

- The way how the CAPEC and related CWE taxonomies are to be used by the evaluator, which needs to consider all the applicable attack patterns and be able to exploit all the related software weaknesses while performing the subsequent AVA_VAN activities.

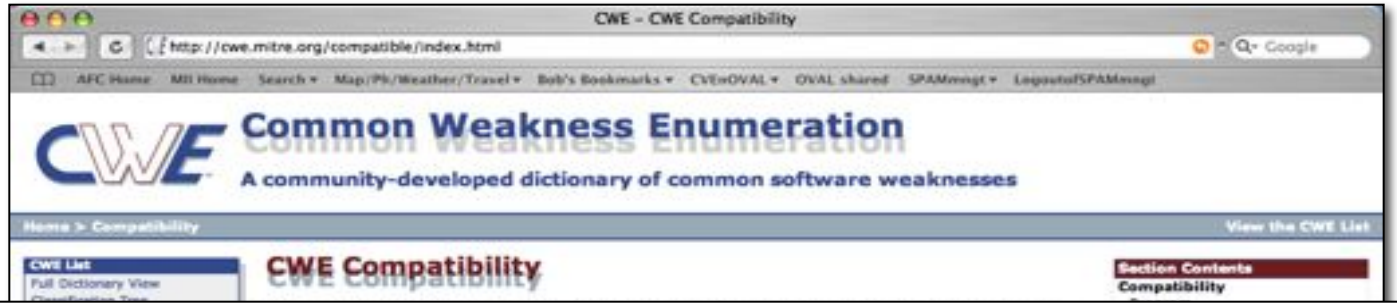
- How incomplete entries from the CAPEC are to be addressed during an evaluation.

- How to incorporate to the evaluation attacks and weaknesses not included in the CAPEC.

© 2012 MITRE

CWE Compatibility & Effectiveness Program

(launched Feb 2007)



SECURITY DATABASE
KEEP A VIGILANT EYE ON YOUR DEFENSES
KDM Analytics
VERACODE
FORTIFY SOFTWARE
armorize technologies
Secure Your Web Code
SecurityReason
Our Reason is Security
Astyran Pte Ltd
coverity
PARASOFT
SkillBridge
Klocwork
EMC²
CODENOMICON
defensics
(ISC)²
CERIAS
CENZIC
GRAMMATECH
symantec
SANS
IPA
INFORMATION-TECHNOLOGY PROMOTION AGENCY, JAPAN
SofCheck
IBM
watchfire
checkmarx
OUNCE LABS
EC-Council
PRQA
Programming Research
CAST
LDRA
Software Technology
cigital
hp
SPI DYNAMICS

cwe.mitre.org/compatible/

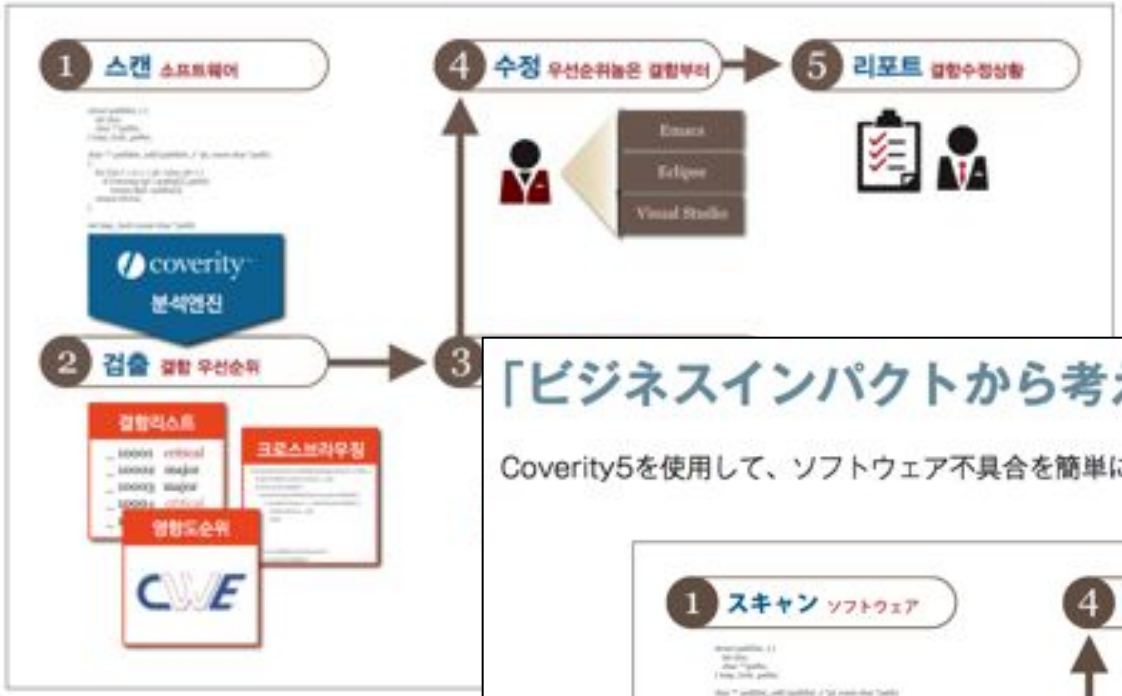
First 13 CWE Compatible Certificates Awarded 28 Feb 2012

© 2012 MITRE


armorize
The Web Malware Experts
We speak
CWE
We speak
CVE
SANS
GIAC
We speak
CWE
cwe.mitre.org
int val; };
my_struct *ptr, int x, int y)
) && (x < y) || (ptr->val > 0))
ptr->val;

[비즈니스 임팩트를 줄여주는 새로운 품질 관리 방법론]

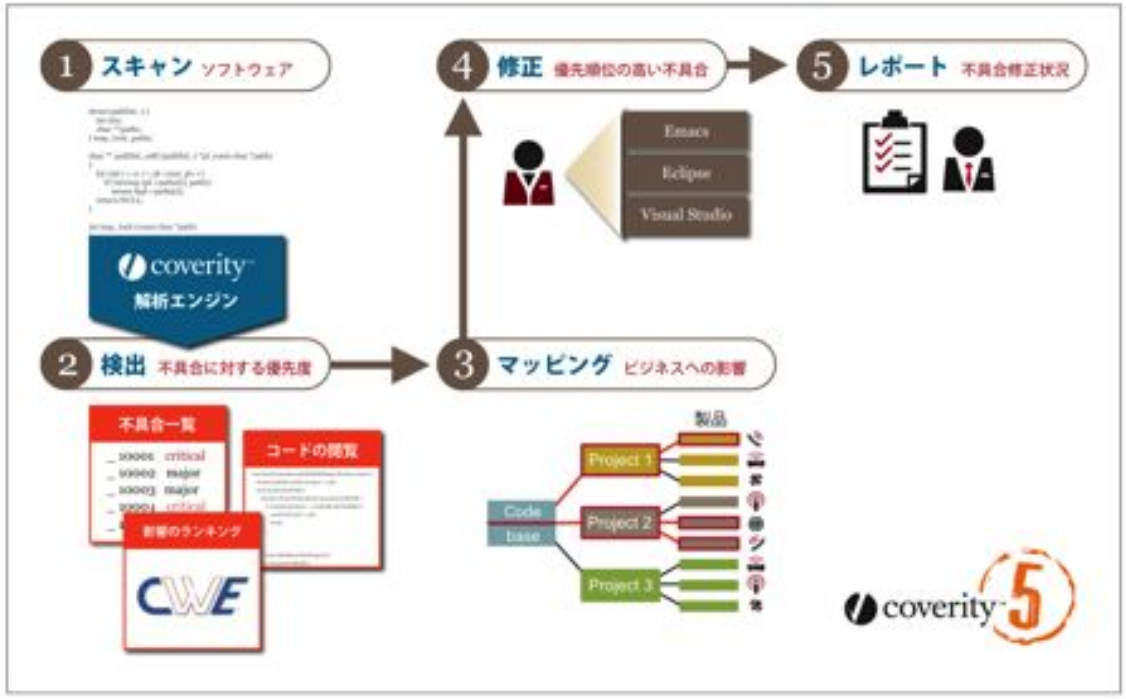
y5를 사용하여, 소프트웨어 결함을 없애는 5가지 스텝은 아래와 같습니다.



Korean

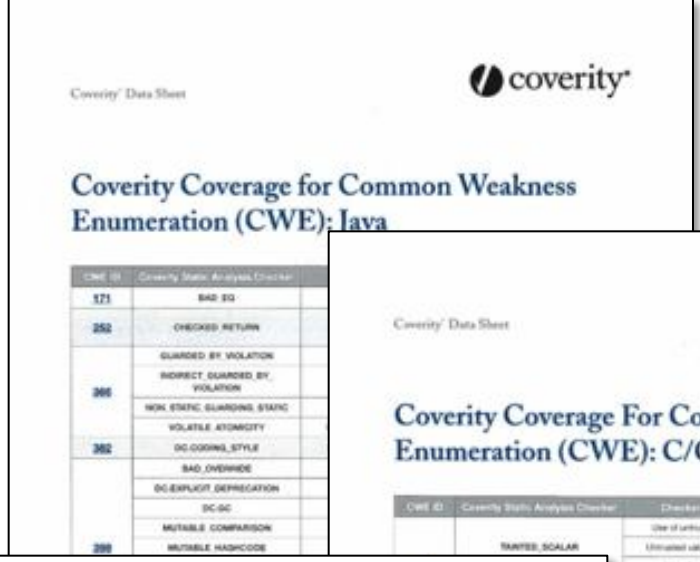
「ビジネスインパクトから考える新しい品質管理」

Coverity5を使用して、ソフトウェア不具合を簡単に除去する 5ステップは以下の通りです。



Japanese

CWE Coverage – Implemented…

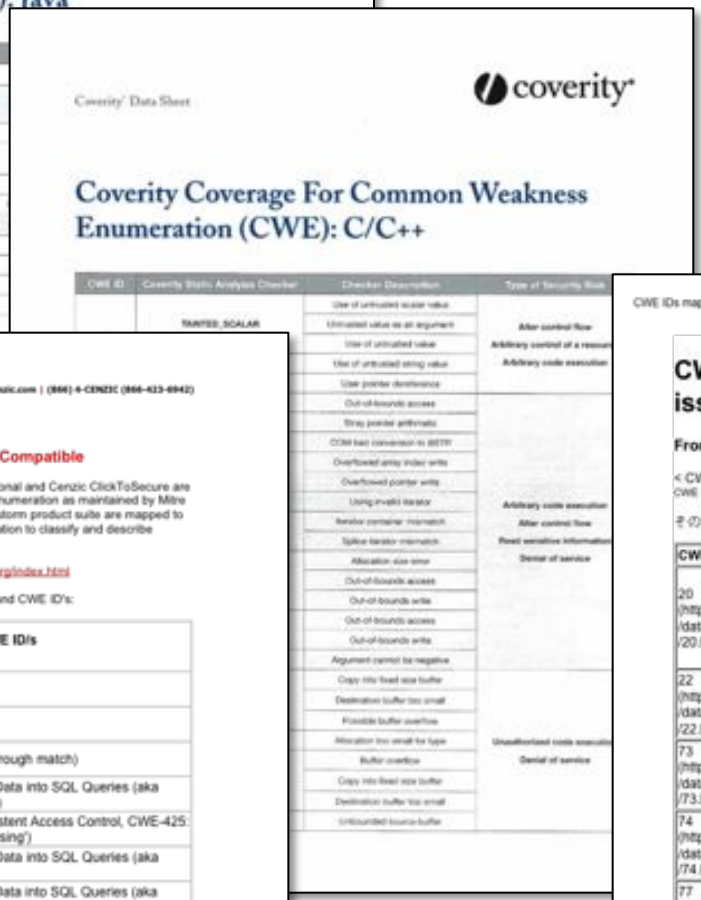


Coverity Data Sheet

coverity

Coverity Coverage for Common Weakness Enumeration (CWE): Java

CWE ID	Coverity Static Analysis Checker
171	BAD_EQ
252	CHECKED_RETURN
366	GUARDED_BY_VIOLATION
	INDIRECT_GUARDED_BY_VIOLATION
	NON_STATIC_GUARDING_STATIC
	VOLATILE_ATOMICITY
382	OC.CODING_STYLE
	BAD_OVERRIDE
	OC.EXPLICIT_DEPRECATION
	OC.GC
	MUTABLE_COMPARISON
200	MUTABLE_HASHCODE

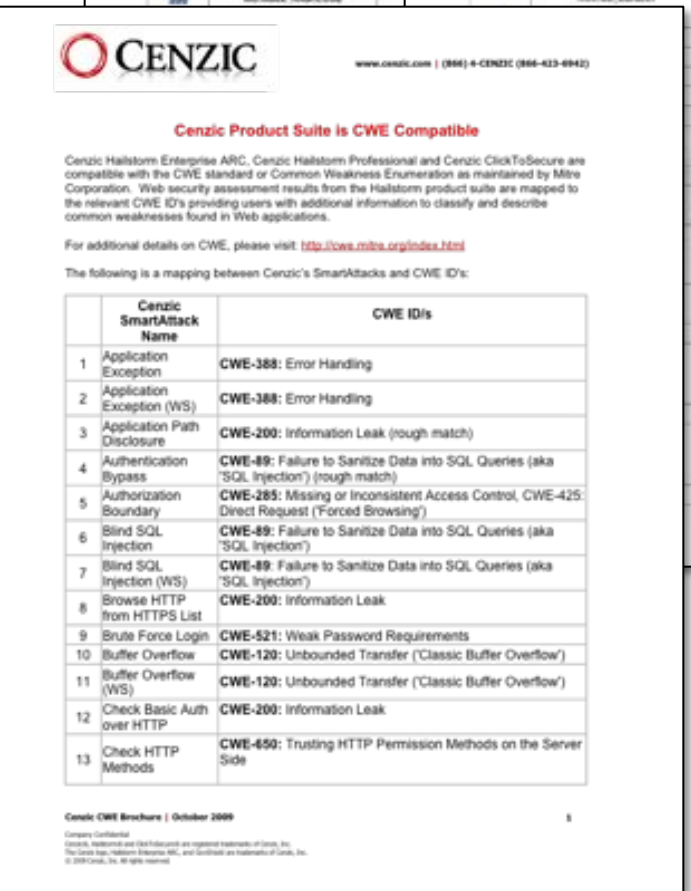


Coverity Data Sheet

coverity

Coverity Coverage For Common Weakness Enumeration (CWE): C/C++

CWE ID	Coverity Static Analysis Checker	Checker Description	Type of Security Risk
	TAINTED_SCALAR	Use of untrusted scalar value	After control flow
		Untrusted value as an argument	Arbitrary control of a resource
		Use of untrusted value	Arbitrary code execution
		Use of untrusted string value	
		User pointer dereference	
		Out-of-bounds access	
		Stray pointer arithmetic	
		COM bad conversion to BSTR	
		Overflowed array index write	
		Overflowed pointer write	
		Using invalid iterator	Arbitrary code execution
		Iterator container mismatch	After control flow
		Splice iterator mismatch	Read sensitive information
		Allocation size error	Denial of service
		Out-of-bounds access	
		Out-of-bounds write	
		Out-of-bounds access	
		Out-of-bounds write	
		Argument cannot be negative	
		Copy into fixed size buffer	
		Destination buffer too small	
		Possible buffer overflow	
		Allocation too small for type	Unauthorized code execution
		Buffer overflow	Denial of service
		Copy into fixed size buffer	
		Destination buffer too small	
		Unbounded source buffer	



CENZIC

www.cenzic.com | (866) 4-CENZIC (866-423-6942)

Cenzic Product Suite is CWE Compatible

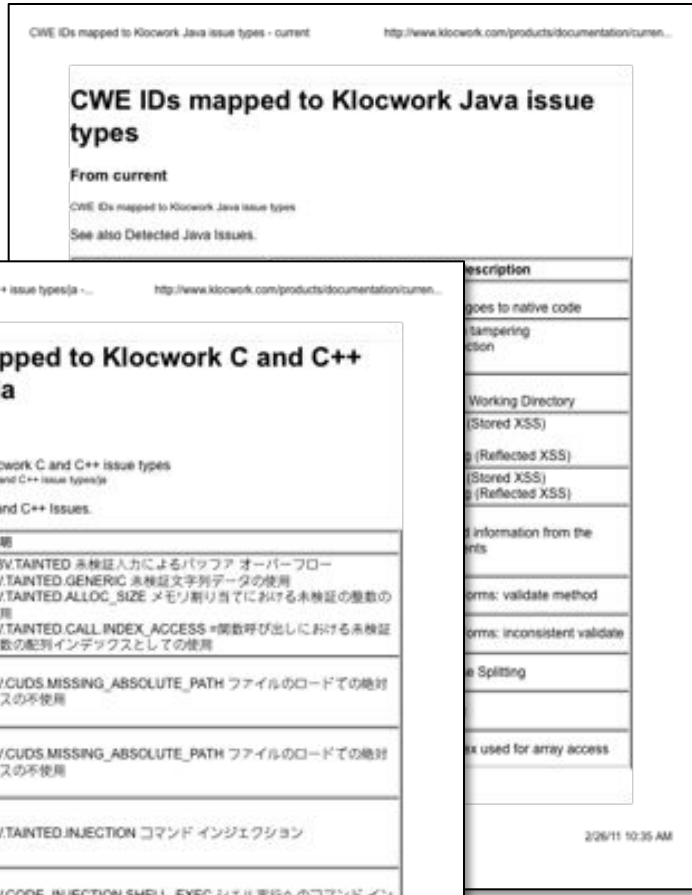
Cenzic Hailstorm Enterprise ARC, Cenzic Hailstorm Professional and Cenzic ClickToSecure are compatible with the CWE standard or Common Weakness Enumeration as maintained by Mitre Corporation. Web security assessment results from the Hailstorm product suite are mapped to the relevant CWE ID's providing users with additional information to classify and describe common weaknesses found in Web applications.

For additional details on CWE, please visit: http://cwe.mitre.org/index.html

The following is a mapping between Cenzic's SmartAttacks and CWE ID's:

	Cenzic SmartAttack Name	CWE ID/s
1	Application Exception	**CWE-388:** Error Handling
2	Application Exception (WS)	**CWE-388:** Error Handling
3	Application Path Disclosure	**CWE-200:** Information Leak (rough match)
4	Authentication Bypass	**CWE-89:** Failure to Sanitize Data into SQL Queries (aka 'SQL Injection') (rough match)
5	Authorization Boundary	**CWE-285:** Missing or Inconsistent Access Control, CWE-425: Direct Request ('Forced Browsing')
6	Blind SQL Injection	**CWE-89:** Failure to Sanitize Data into SQL Queries (aka 'SQL Injection')
7	Blind SQL Injection (WS)	**CWE-89**: Failure to Sanitize Data into SQL Queries (aka 'SQL Injection')
8	Browse HTTP from HTTPS List	**CWE-200:** Information Leak
9	Brute Force Login	**CWE-521:** Weak Password Requirements
10	Buffer Overflow	**CWE-120:** Unbounded Transfer ('Classic Buffer Overflow')
11	Buffer Overflow (WS)	**CWE-120:** Unbounded Transfer ('Classic Buffer Overflow')
12	Check Basic Auth over HTTP	**CWE-200:** Information Leak
13	Check HTTP Methods	**CWE-650:** Trusting HTTP Permission Methods on the Server Side

Cenzic CWE Brochure | October 2009 1



CWE IDs mapped to Klocwork Java issue types - current http://www.klocwork.com/products/documentation/curren...

CWE IDs mapped to Klocwork Java issue types

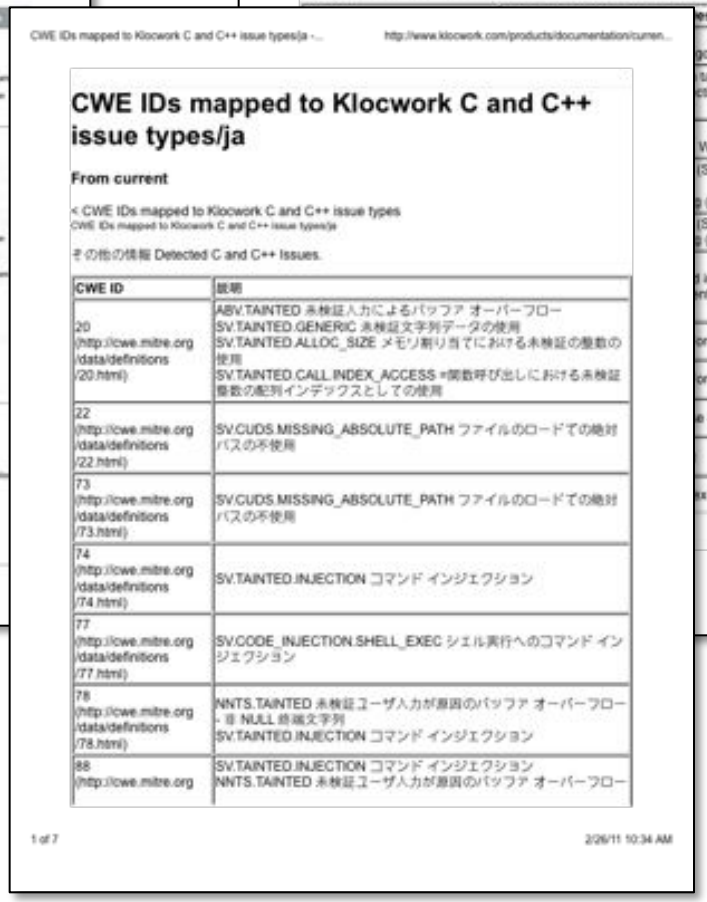
From current

CWE IDs mapped to Klocwork Java issue types

See also Detected Java Issues.

escription
goes to native code
tampering ction
Working Directory
(Stored XSS)
(Reflected XSS)
(Stored XSS)
(Reflected XSS)
information from the ints
orms: validate method
orms: inconsistent validate
e Splitting
ix used for array access

2/26/11 10:35 AM



CWE IDs mapped to Klocwork C and C++ issue types/ja -... http://www.klocwork.com/products/documentation/curren...

CWE IDs mapped to Klocwork C and C++ issue types/ja

From current

< CWE IDs mapped to Klocwork C and C++ issue types

CWE IDs mapped to Klocwork C and C++ issue types/ja

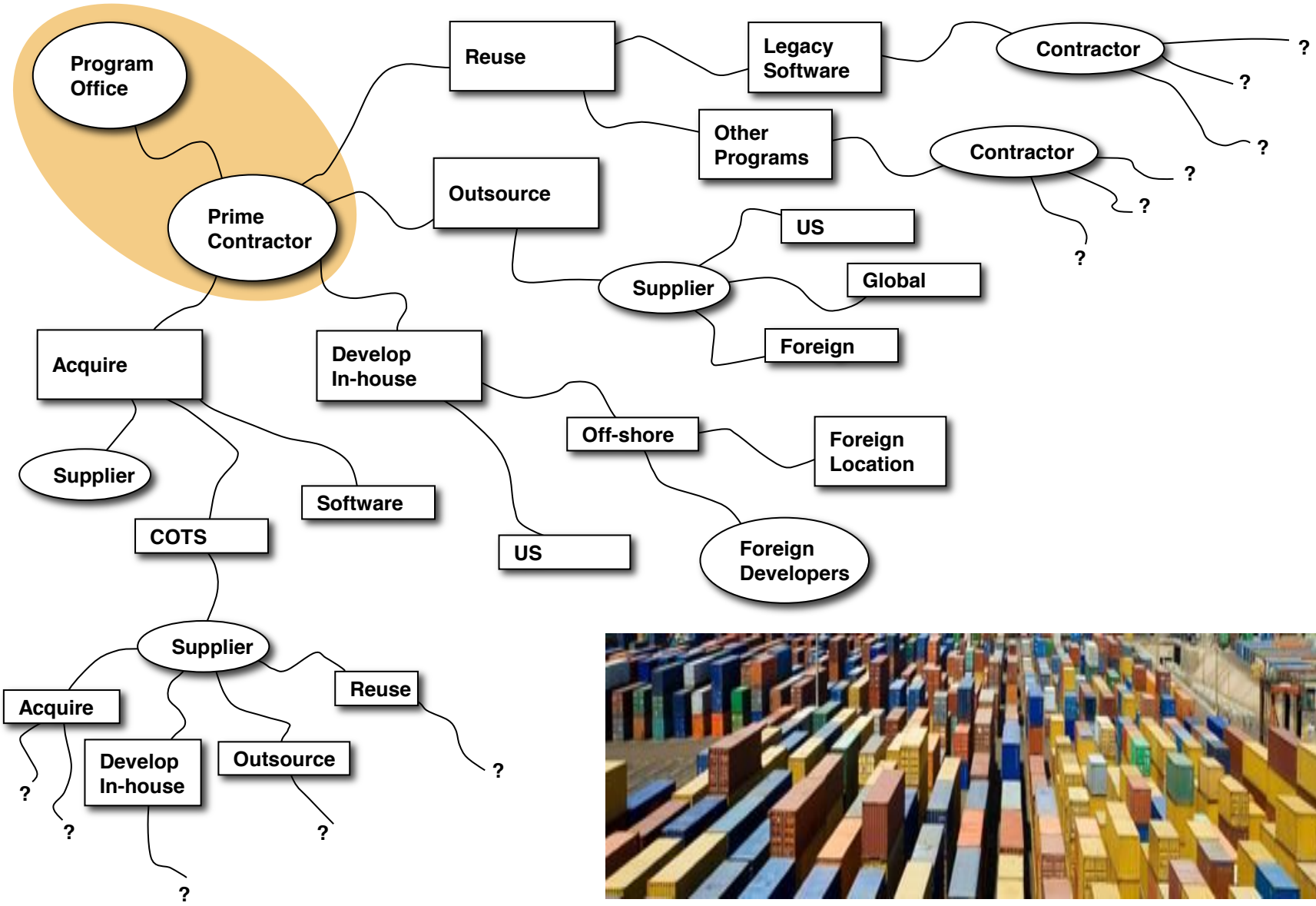
その他の情報 Detected C and C++ Issues.

CWE ID	説明
20 (http://cwe.mitre.org/data/definitions/20.html)	ABV.TAINTED 未検証入力によるバッファ オーバーフロー SV.TAINTED.GENERIC 未検証文字列データの使用 SV.TAINTED.ALLOC_SIZE メモリ割り当てにおける未検証の整数の使用 SV.TAINTED.CALL.INDEX_ACCESS =関数呼び出しにおける未検証整数の配列インデックスとしての使用
22 (http://cwe.mitre.org/data/definitions/22.html)	SV.CUDS.MISSING_ABSOLUTE_PATH ファイルのロードでの絶対パスの不使用
73 (http://cwe.mitre.org/data/definitions/73.html)	SV.CUDS.MISSING_ABSOLUTE_PATH ファイルのロードでの絶対パスの不使用
74 (http://cwe.mitre.org/data/definitions/74.html)	SV.TAINTED.INJECTION コマンド インジェクション
77 (http://cwe.mitre.org/data/definitions/77.html)	SV.CODE_INJECTION.SHELL_EXEC シェル実行へのコマンド インジェクション
78 (http://cwe.mitre.org/data/definitions/78.html)	NNTS.TAINTED 未検証ユーザ入力が原因のバッファ オーバーフロー - ※ NULL 終端文字列 SV.TAINTED.INJECTION コマンド インジェクション
88 (http://cwe.mitre.org	SV.TAINTED.INJECTION コマンド インジェクション NNTS.TAINTED 未検証ユーザ入力が原因のバッファ オーバーフロー

1 of 7 2/26/11 10:34 AM

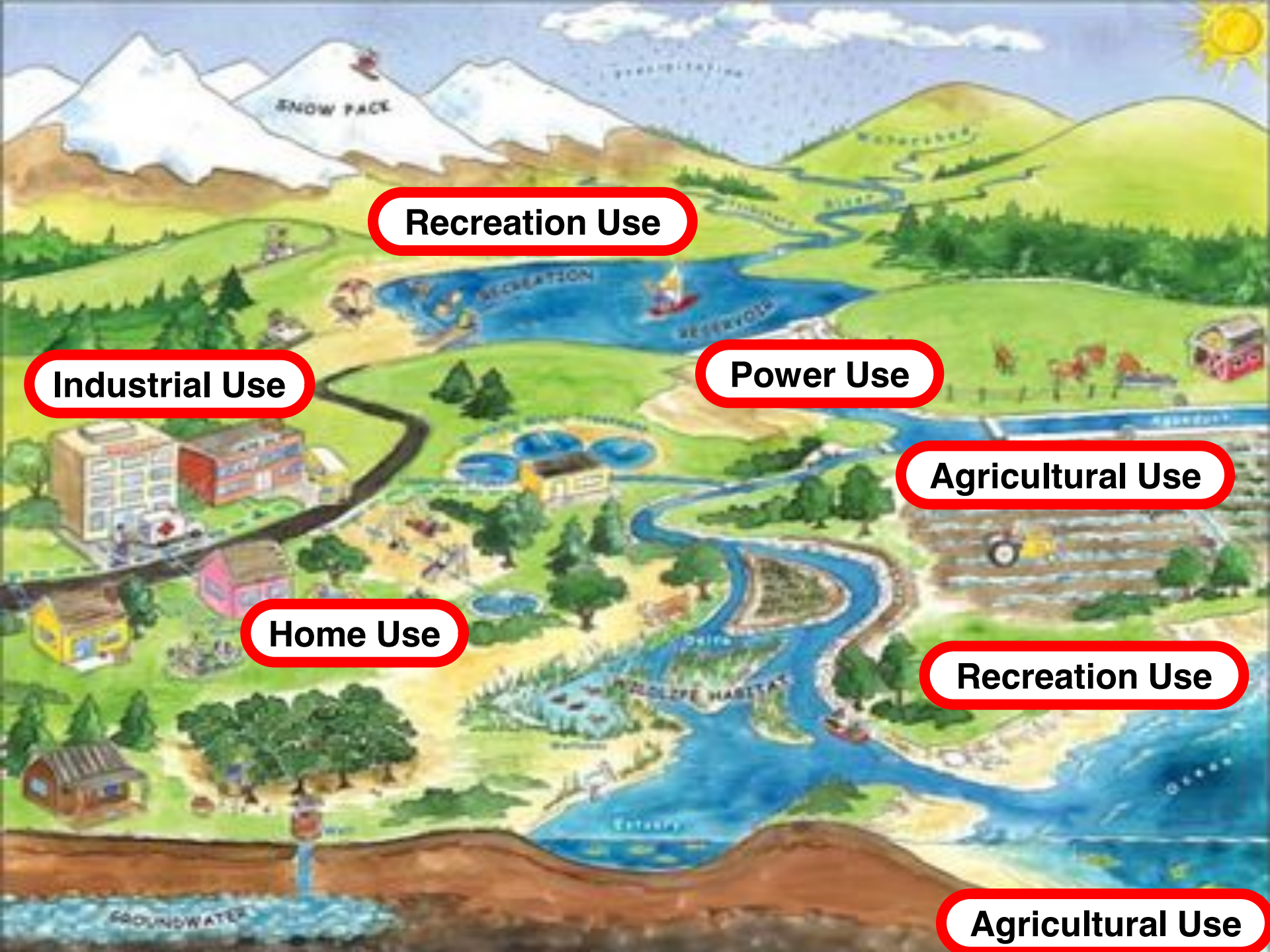
© 2012 MITRE

The Software Supply Chain



* "Scope of Supplier Expansion and Foreign Involvement" graphic in DACS www.softwaretechnews.com Secure Software Engineering, July 2005 article "Software Development Security: A Risk Management Perspective" synopsis of May 2004 GAO-04-678 report "Defense Acquisition: Knowledge of Software Suppliers Needed to Manage Risks"

© 2012 MITRE


Recreation Use
Industrial Use
Power Use
Agricultural Use
Home Use
Recreation Use
Agricultural Use


Too much water

Prioritizing weaknesses to be mitigated

OWASP
The Open Web Application Security Project
http://www.owasp.org

OWASP Top 10

CWE/SANS Top 25

Lists are a good start but they are designed to be broadly applicable

We would like a way to specify priorities based on business/mission risk

© 2012 MITRE

Common Weakness Risk Analysis Framework (CWRAF)

How do I identify which of the 800+ CWE's are most important for my specific business domain, technologies and environment?

Common Weakness Scoring System (CWSS)

How do I rank the CWE's I care about according to my specific business domain, technologies and environment?

How do I identify and score weaknesses important to my organization?

© 2012 MITRE

CWRAF-Level Technical Impacts

1. Modify data
2. Read data
3. DoS: unreliable execution
4. DoS: resource consumption
5. Execute unauthorized code or commands
6. Gain privileges / assume identity
7. Bypass protection mechanism
8. Hide activities

© 2012 MITRE

Common Weakness Risk Analysis Framework (CWRAF)

Technical Impacts

1. Modify data
2. Read data
3. DoS: unreliable execution
4. DoS: resource consumption
5. Execute unauthorized code or commands
6. Gain privileges / assume identity
7. Bypass protection mechanism
8. Hide activities

Weightings

W1=0
W2=0
W3=1
0
W4=4
W5=1
0
W6=0
W7=0
W8=0

×

Layers

1. System
2. Application
3. Network
4. Enterprise

=

Technical Impact Scorecard

Multiple pieces – we'll focus on "Vignettes"

© 2012 MITRE

CWRAF: Technical Impact Scorecard

and each technical impact

	MD	RD	UE	RC	EA	GP	BP	HA
Application								
System		8						
Network								
Enterprise								3

For each layer

assign a weighting from 0 to 10

© 2012 MITRE

CWRAF: Technical Impact Scorecard

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	2
System	8	8	4	2	10	9	5	1
Network	9	5	6	2	10	5	7	1
Enterprise	4	7	6	2	10	6	4	3

These weightings can now be used to evaluate individual CWE's based on each CWE's Technical Impacts

Note: Values for illustrative purposes only

© 2012 MITRE

Note: Values for illustrative purposes only

Notional

	MD	RD	UE	RC	EA	GP	BP	HA
Application	9	7	3	2	10	8	7	2
System	8	8	4	2	10	9	5	1
Network	9	5	6	2	10	5	7	1
Enterprise	4	7	6	2	10	6	4	3

CWE-78 Technical Impacts

CWSS Formula

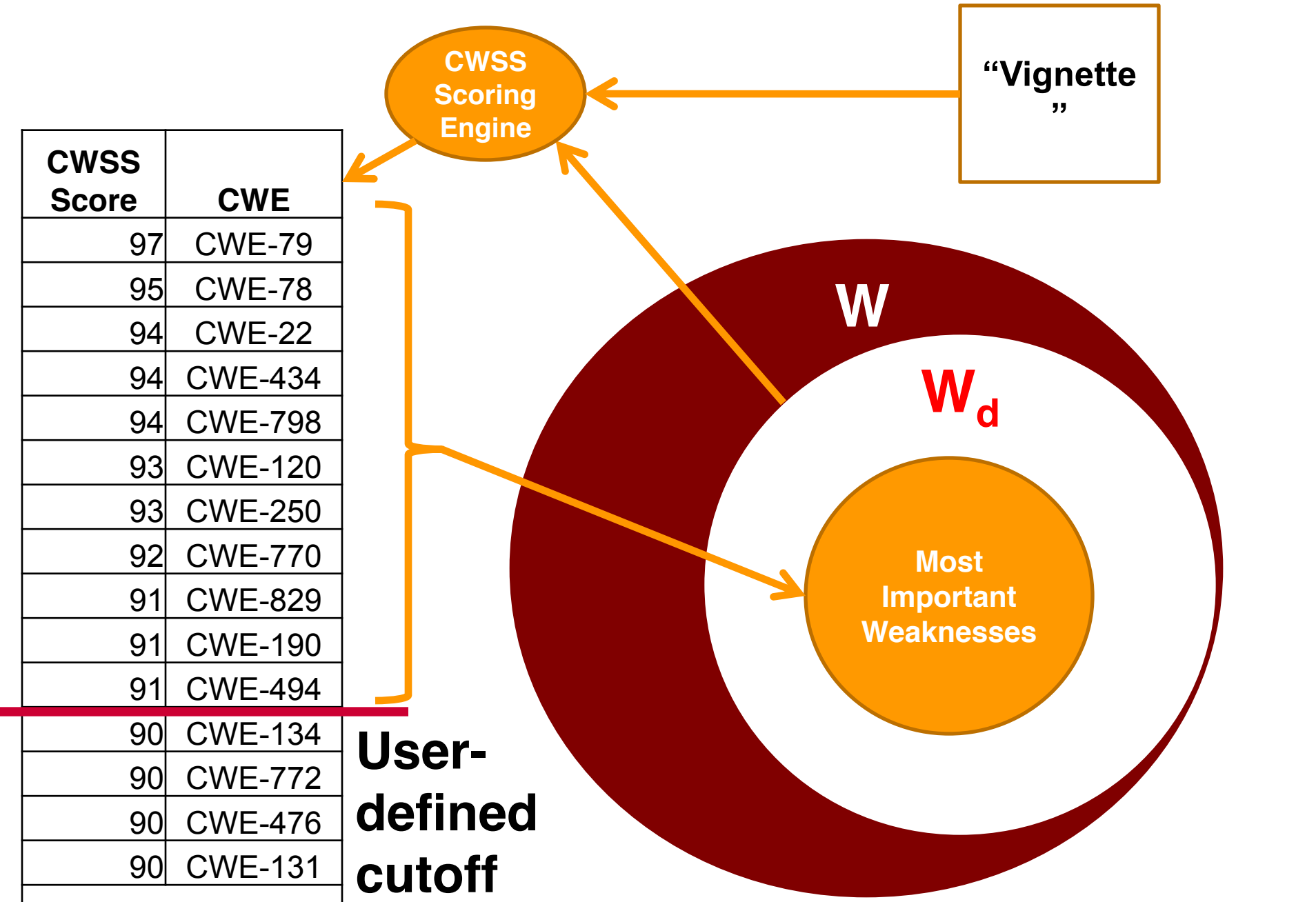
= **95**

CWSS Score for CWE-78 *for this vignette*

Common Weakness Scoring System (CWSS)

© 2012 MITRE

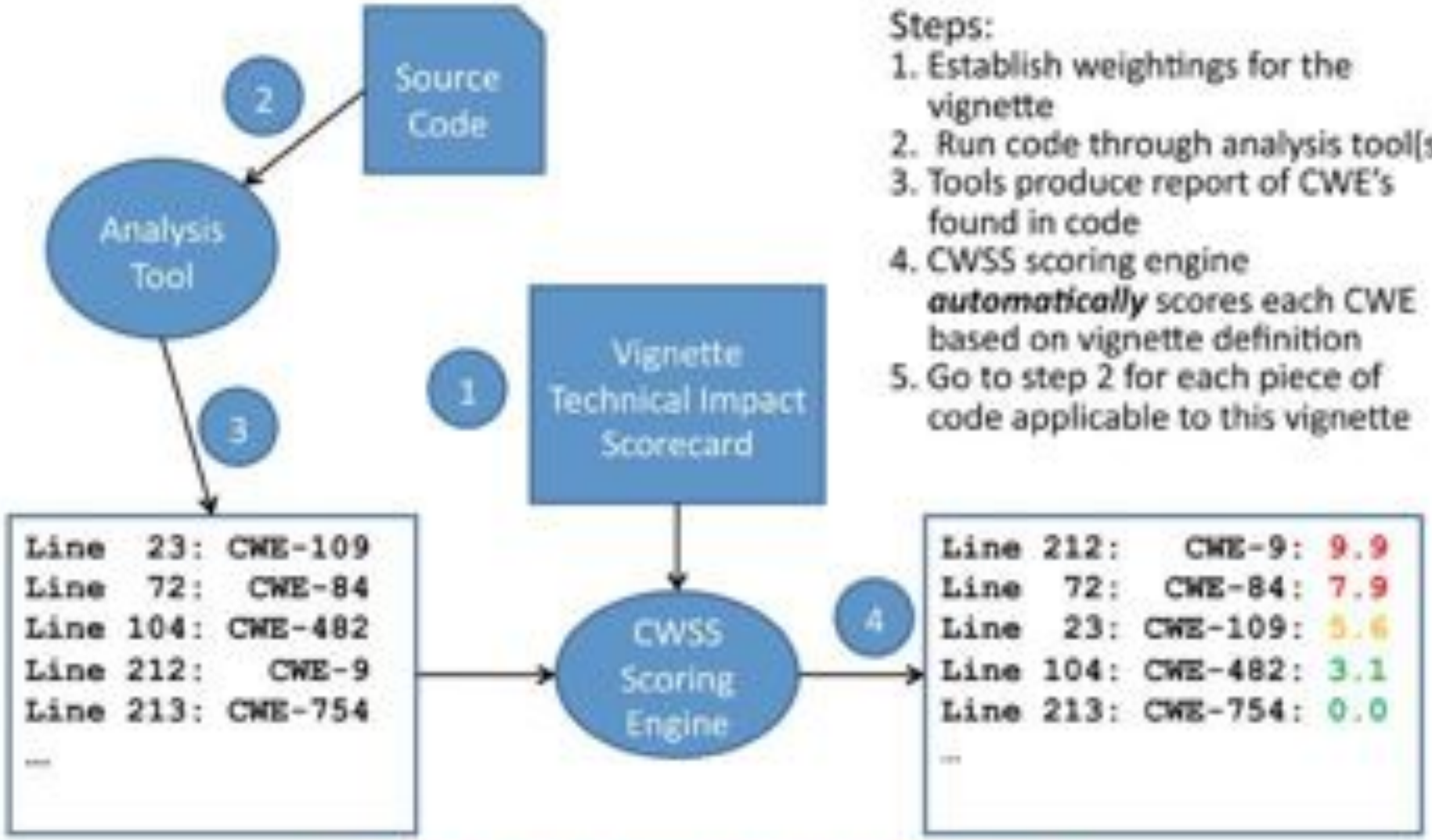
CWSS Score	CWE
97	CWE-79
95	CWE-78
94	CWE-22
94	CWE-434
94	CWE-798
93	CWE-120
93	CWE-250
92	CWE-770
91	CWE-829
91	CWE-190
91	CWE-494
90	CWE-134
90	CWE-772
90	CWE-476
90	CWE-131
...	



CWRAF/CWSS in a Nutshell

© 2012 MITRE

Scoring Weaknesses Discovered in Code using CWSS



Steps:

1. Establish weightings for the vignette
2. Run code through analysis tool(s)
3. Tools produce report of CWE's found in code
4. CWSS scoring engine ***automatically*** scores each CWE based on vignette definition
5. Go to step 2 for each piece of code applicable to this vignette

Step 1 is only done once – the rest is automatic

Homeland Security

CWSS

MITRE

Organizations that have declared plans to support CWSS in their future offerings and are working with MITRE to help evolve CWSS to meet their customer's and the community's needs for a scoring system for software errors.

GRAMMATECH

Klocwork

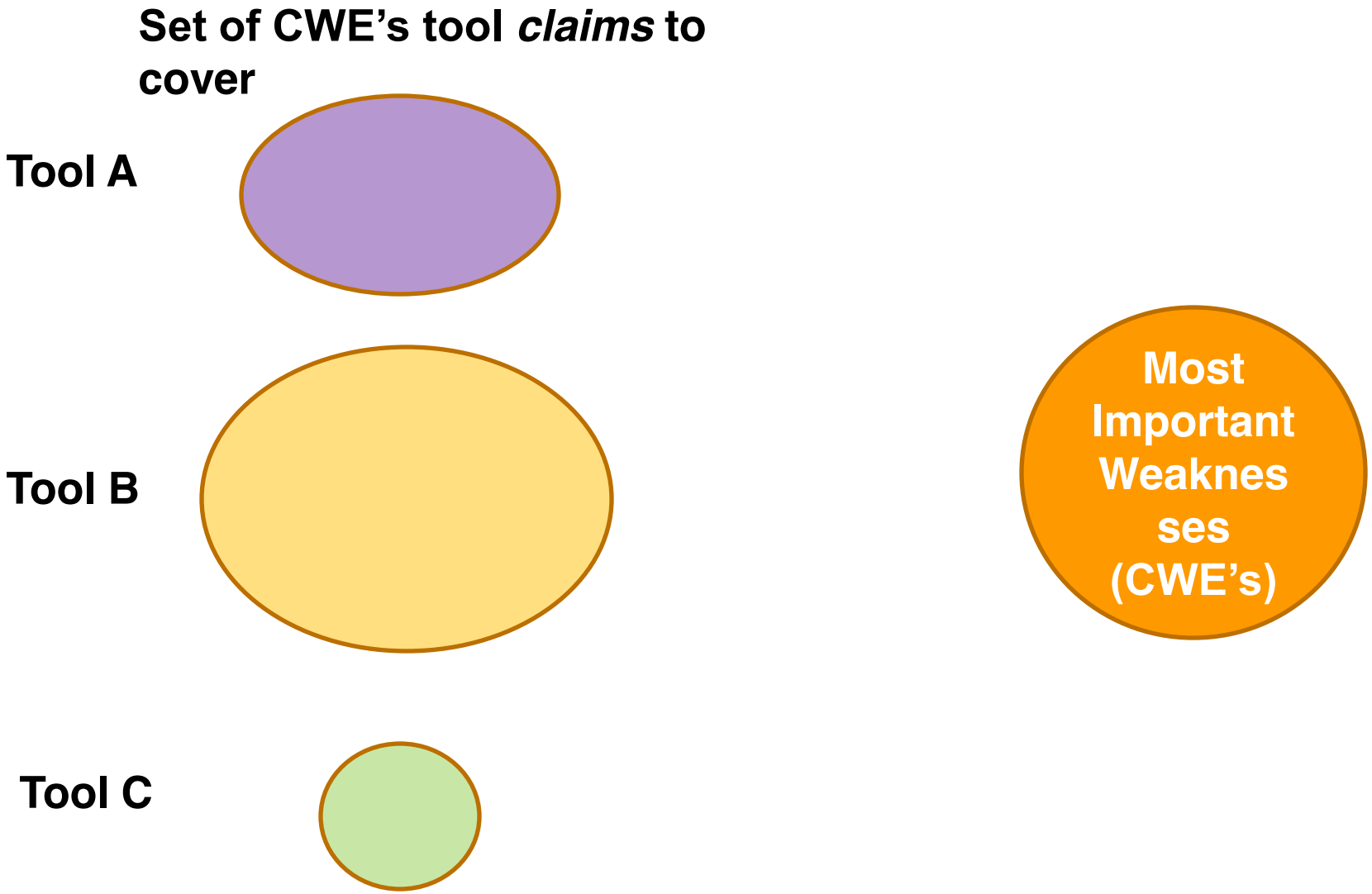
VERACODE

FORTIFY SOFTWARE

SANS

© 2012 MITRE

CWE Coverage Claims Representation



Which static analysis tools find the CWE's I care about?

© 2012 MITRE

CWSS for a Technology Group

50%	Web Vignette 1 … TI(1), TI(2), TI(3),…	Top N List 1
10%	Web Vignette 2 … TI(1), TI(2), TI(3),…	Top N List 2
10%	Web Vignette 3 … TI(1), TI(2), TI(3),…	Top N List 3
10%	Web Vignette 4 … TI(1), TI(2), TI(3),…	Top N List 4
15%	Web Vignette 5 … TI(1), TI(2), TI(3),…	Top N List 5
15%	Web Vignette 6 … TI(1), TI(2), TI(3),…	Top N List 6
	Web Application Technology Group	**Top 10 List**

CWE Top 10 List for Web Applications can be used to:

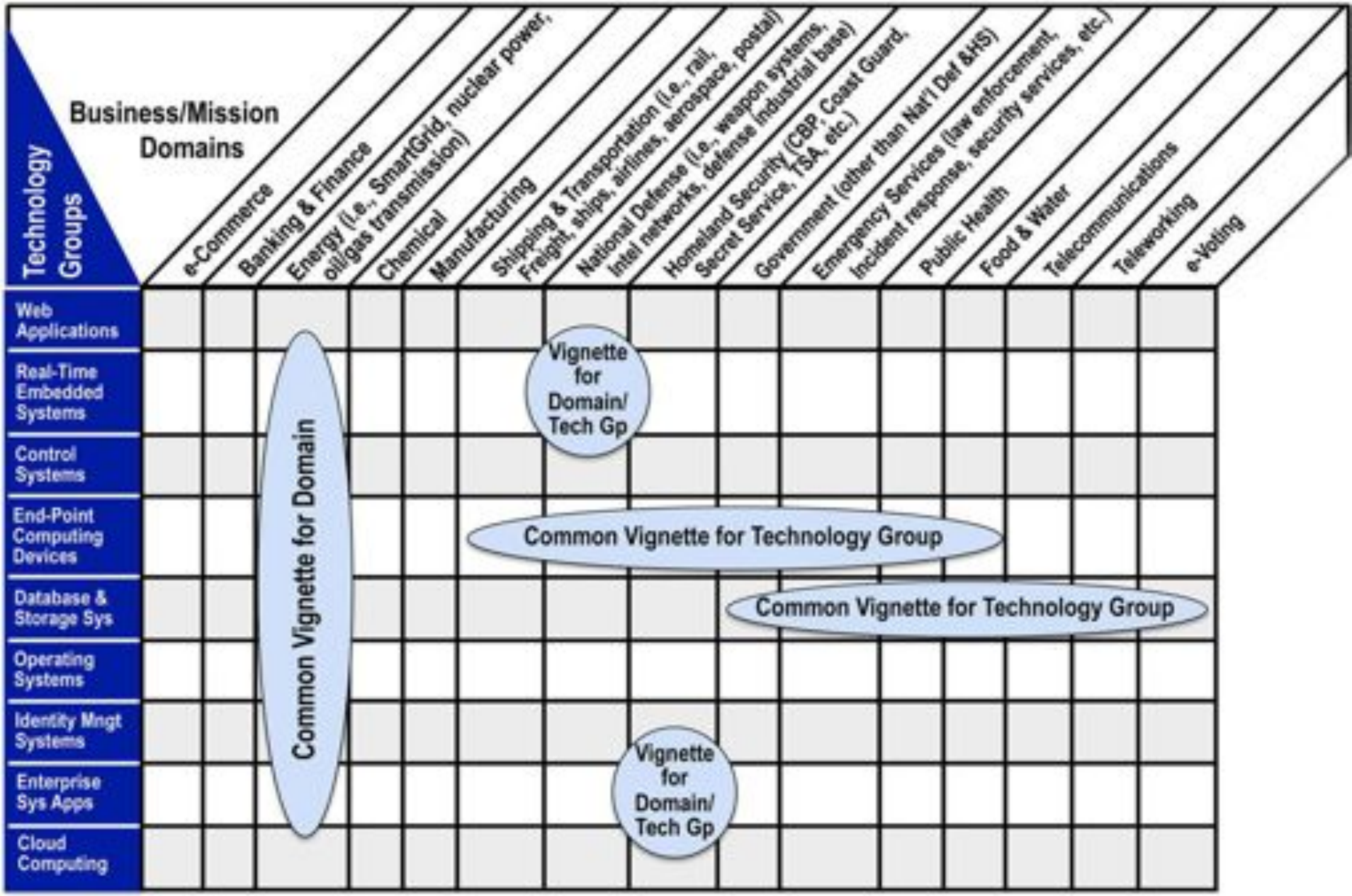
- **Identify skill and training needs for your web team**
- **Include in T's & C's for contracting for web development**
- **Identify tool capability needs to support web assessment**

© 2012 MITRE

Domain Name	Description
E-Commerce	The use of the Internet or other computer networks for the sale of products and services, typically using on-line capabilities.
Banking & Finance	Financial services, including banks, stock exchanges, brokers, investment companies, financial advisors, and government regulatory agencies.
Public Health	Health care, medical encoding and billing, patient information/data, critical or emergency care, medical devices (implantable, partially embedded, patient care), drug development and distribution, food processing, clean water treatment and distribution (including dams and processing facilities), etc.
Energy	Smart Grid (electrical network through a large region, using digital technology for monitoring or control), nuclear power stations, oil and gas transmission, etc.
Chemical	Chemical processing and distribution, etc.
Manufacturing	Plants and distribution channels, supply chain, etc.
Shipping & Transportation	Aerospace systems (such as safety-critical ground aviation systems, on-board avionics, etc), shipping systems, rail systems, etc.
National Security	National security systems (including networks and weapon systems), Defense Industrial Base, etc.
Government and Commercial Security	Homeland Security systems, commercial security systems, etc.
Emergency Services	Systems and services that support first responders, incident management and response, law enforcement, and emergency services for citizens, etc.
Telecommunications	Cellular services, land lines, VOIP, cable & fiber networks, etc.
Telecommuting & Teleworking	Support for employees to have remote access to internal business networks and capabilities.
eVoting	Electronic voting systems, as used within state-run elections, shareholder meetings, etc.

Technology Group	Archetypes/Description
Web Applications	Web browser, web-server, web-based applications and services, etc.
Industrial Control Systems	SCADA, process control system, etc.
Real-time, Embedded Systems	Embedded Device, Programmable logic controller, implanted medical devices, avionics package.
End-point Computing Devices	Smart phone, laptop, personal digital assistant (PDA), and other remote devices that leave the enterprise and/or connect remotely to the enterprise.
Cloud Computing	Hosted applications or capabilities provided over the Internet, including Software-as-a-Service (SaaS), Platform-as-a-Service (PaaS), and Infrastructure as a Service (IaaS).
Operating Systems	General-purpose OS, virtualized OS, Real-time operating system (RTOS), hypervisor, microkernel.
Enterprise Desktop Applications/Systems	Office products such as word processing, spreadsheets, project management, etc.

Vignettes – Technology Groups & Business/Mission Domains



Common Weakness Risk Assessment Framework uses Vignettes with Archetypes to identify top CWEs in respective Domain/Technology Groups

©2012 MITRE

CWRAF

MITRE

Organizations that have declared plans to work on CWRAF Vignettes and Technical Scorecards with MITRE to help evolve CWRAF to meet their customer's and the community's needs for a scoring system for software errors.

Trustwave® SpiderLabs®

DTCC®

CISQ

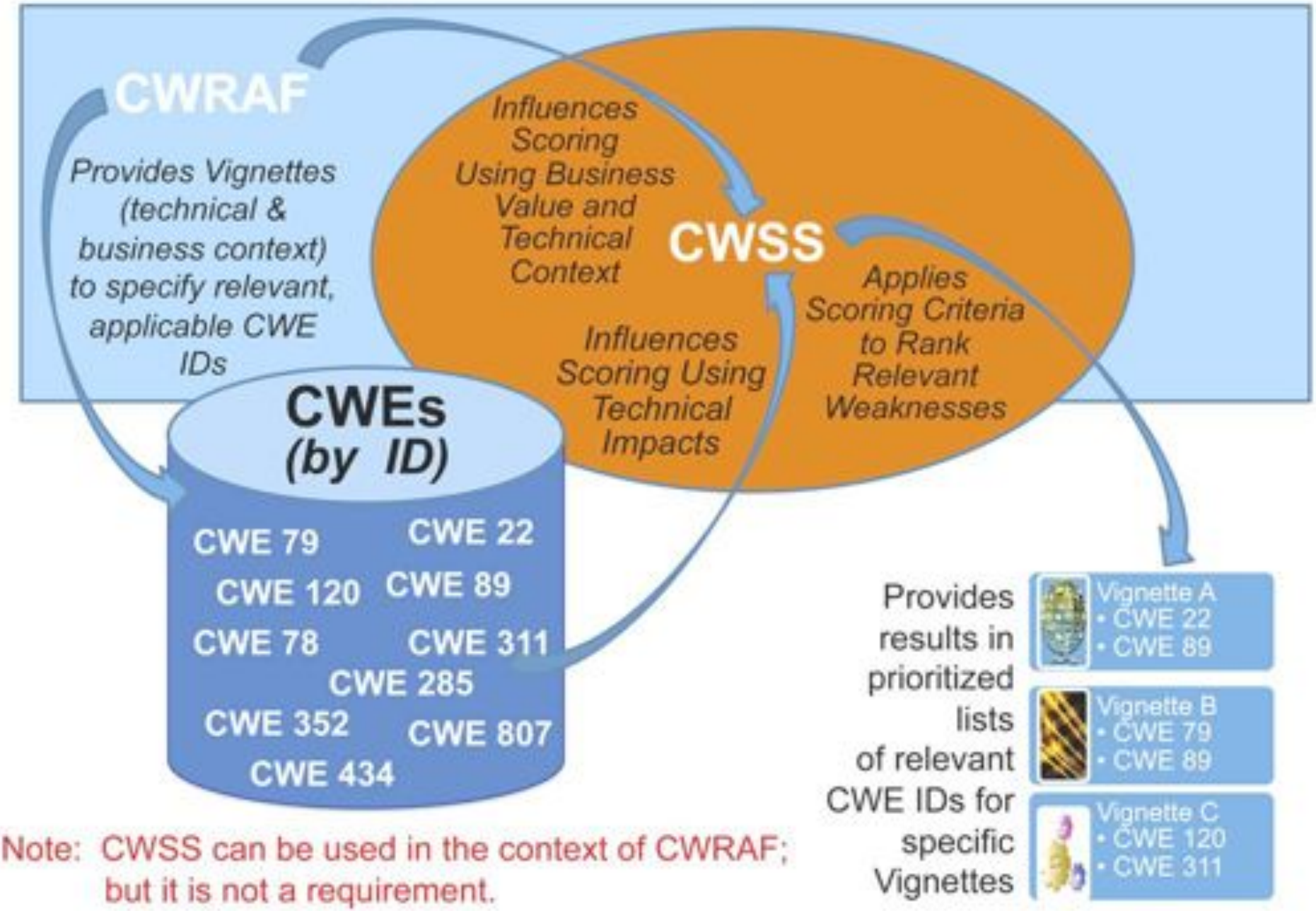
EC-Council

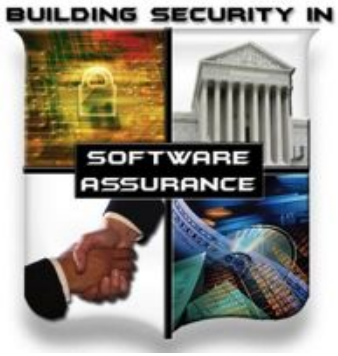
OWASP
The Open Web Application Security Project

SAIC®

© 2012 MITRE

Relationships between CWRAF, CWSS, and CWE





Contact Info

Making Security Measurable™

cwss@mitre.org

cwe@mitre.org

CWE™

CWSS™

CWRAF™

© 2012 MITRE