



“Vendor Neutral Security  
Measurement & Management  
with Standards”

Robert A. Martin  
**MITRE**

ITU-T Workshop on  
**Addressing  
security  
challenges**  
on a global scale



6 - 7 December 2010 Geneva, Switzerland

**As a public interest company, MITRE works in partnership with the government to address issues of critical national importance.**



**MITRE**

# Our Locations

- Principal locations in Bedford, Mass., and McLean, Va., with more than 60 sites worldwide



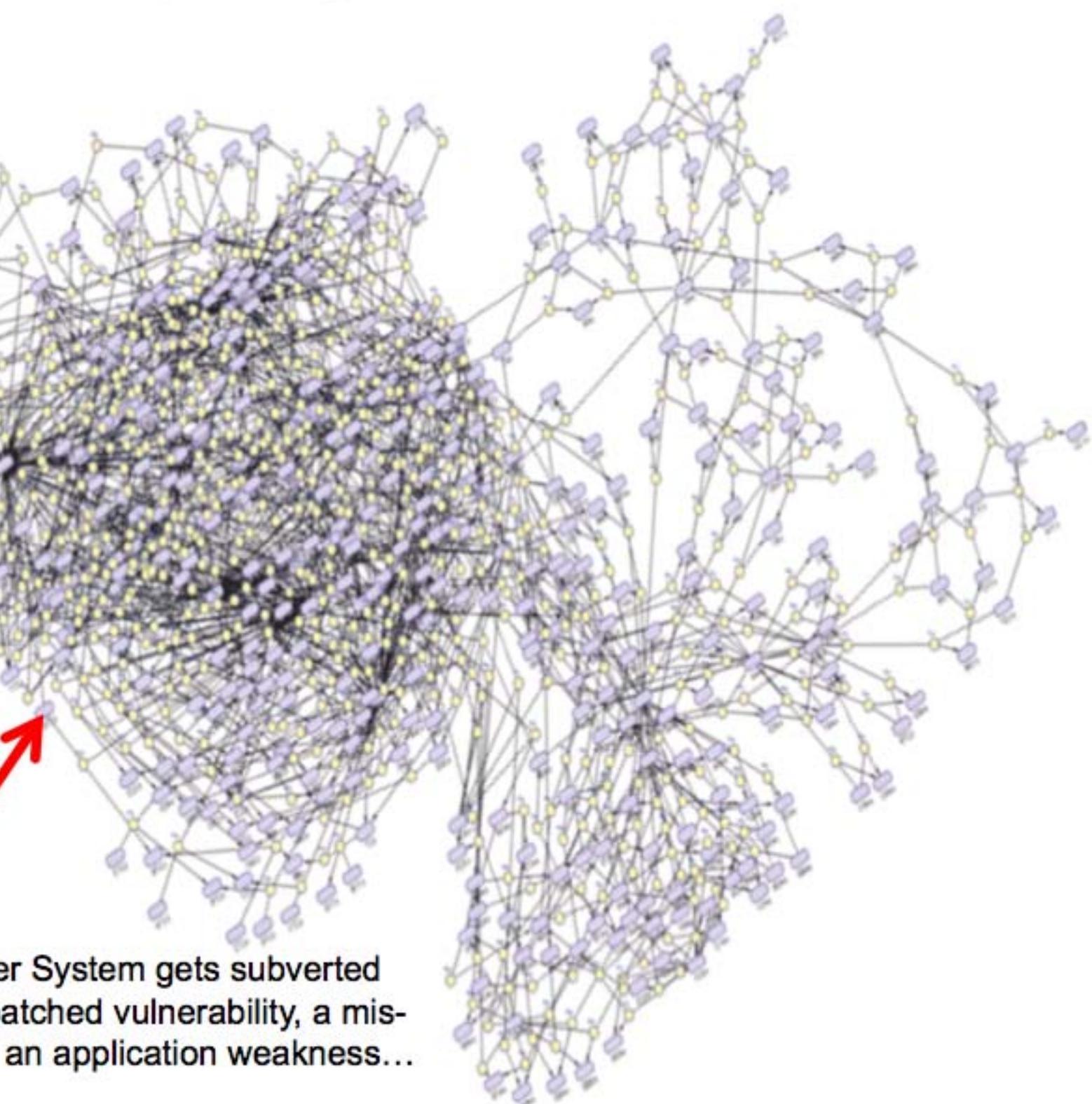
# MITRE's Workforce . . . Our Key Asset

- 7,000 employees worldwide
- Committed to public service
- Technically skilled, highly collaborative
- Work side-by-side with our sponsors



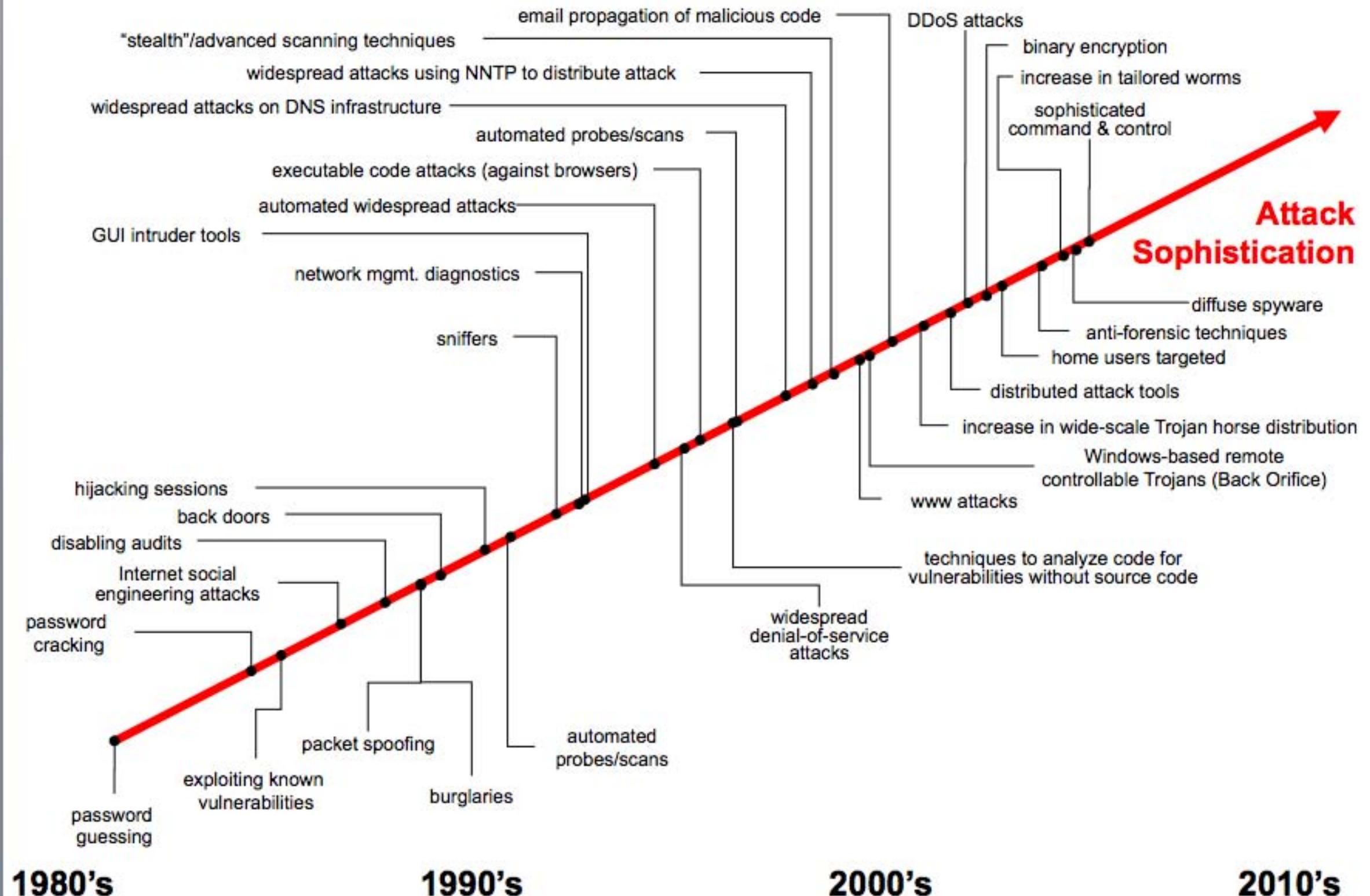
# Today Everything's Connected

Your System is  
attackable...

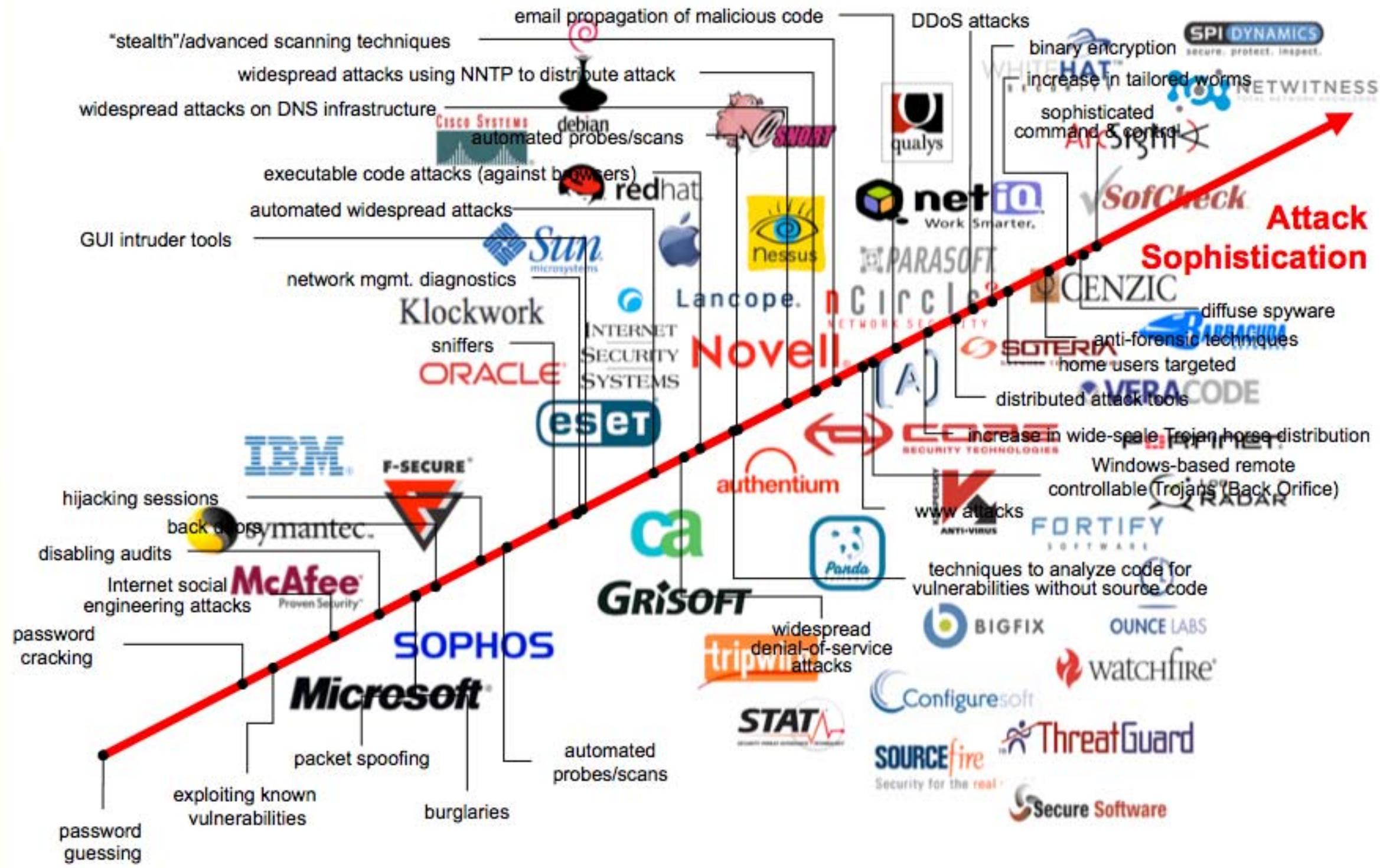


When this Other System gets subverted  
through an un-patched vulnerability, a mis-  
configuration, or an application weakness...

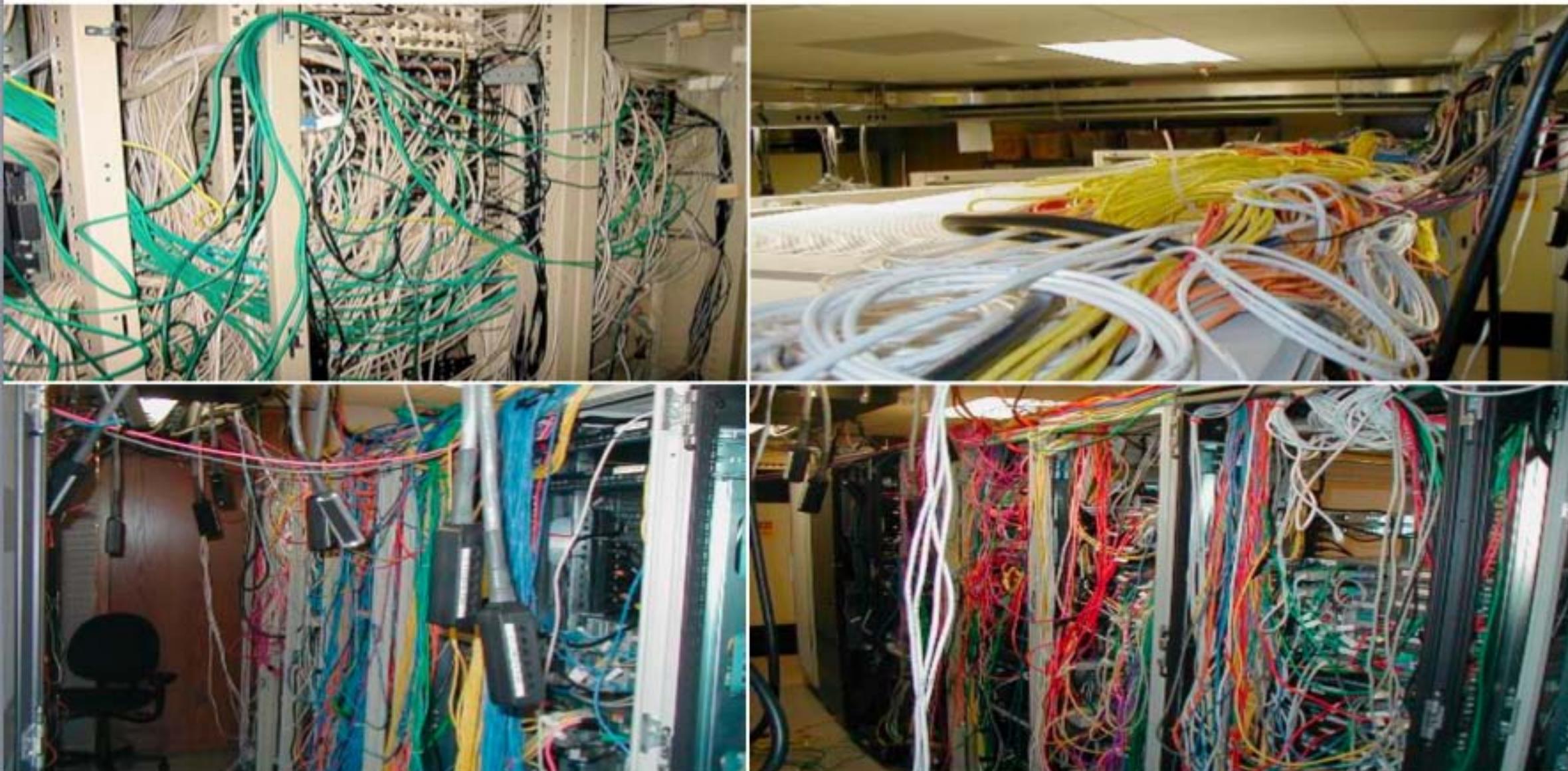
# Cyber Threats Emerged Over Time



# Solutions Also Emerged Over Time



# Like Security - Networks Evolved

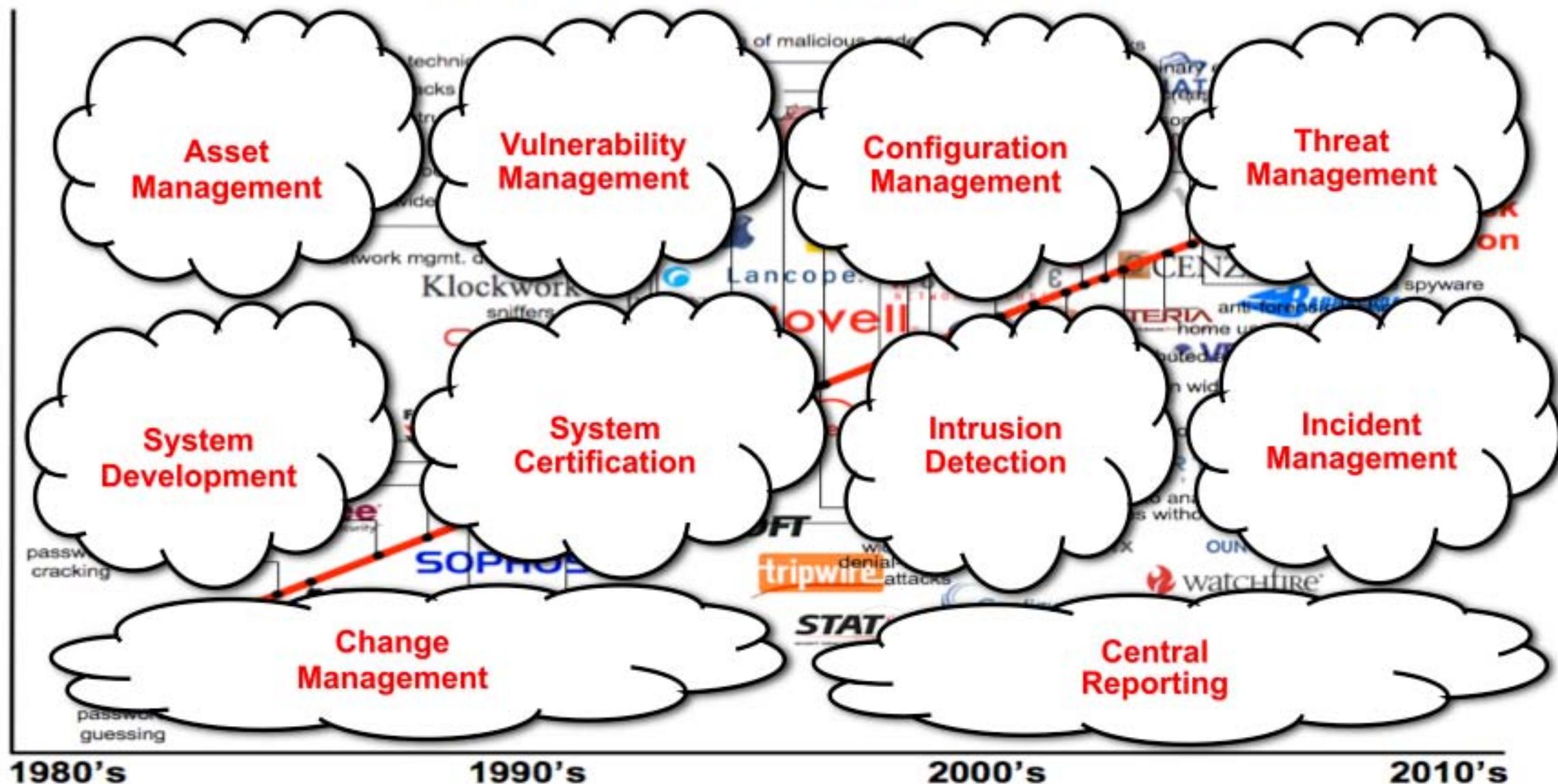


Each new solution had to integrate with the existing solutions  
-->> every enterprise ends up learning as they go and has a  
“unique” tapestry of solutions with “local practices”

**But A More Supportable  
Solution Is Possible with  
Standardized Approaches  
and the application of  
Architecting Principles**



# Architecting Security with Information Standards for COIs



# What Do The Informational Building Blocks for “Architecting Security” Look Like?

- Standard ways for **enumerating** “things we care about”
- **Languages/Formats** for encoding/carrying high fidelity content about the “things we care about”
- **Repositories** of this content for use in communities or individual organizations
- **Adoption/branding and vetting** programs to encourage adoption by tools and services



# The Building Blocks Are:

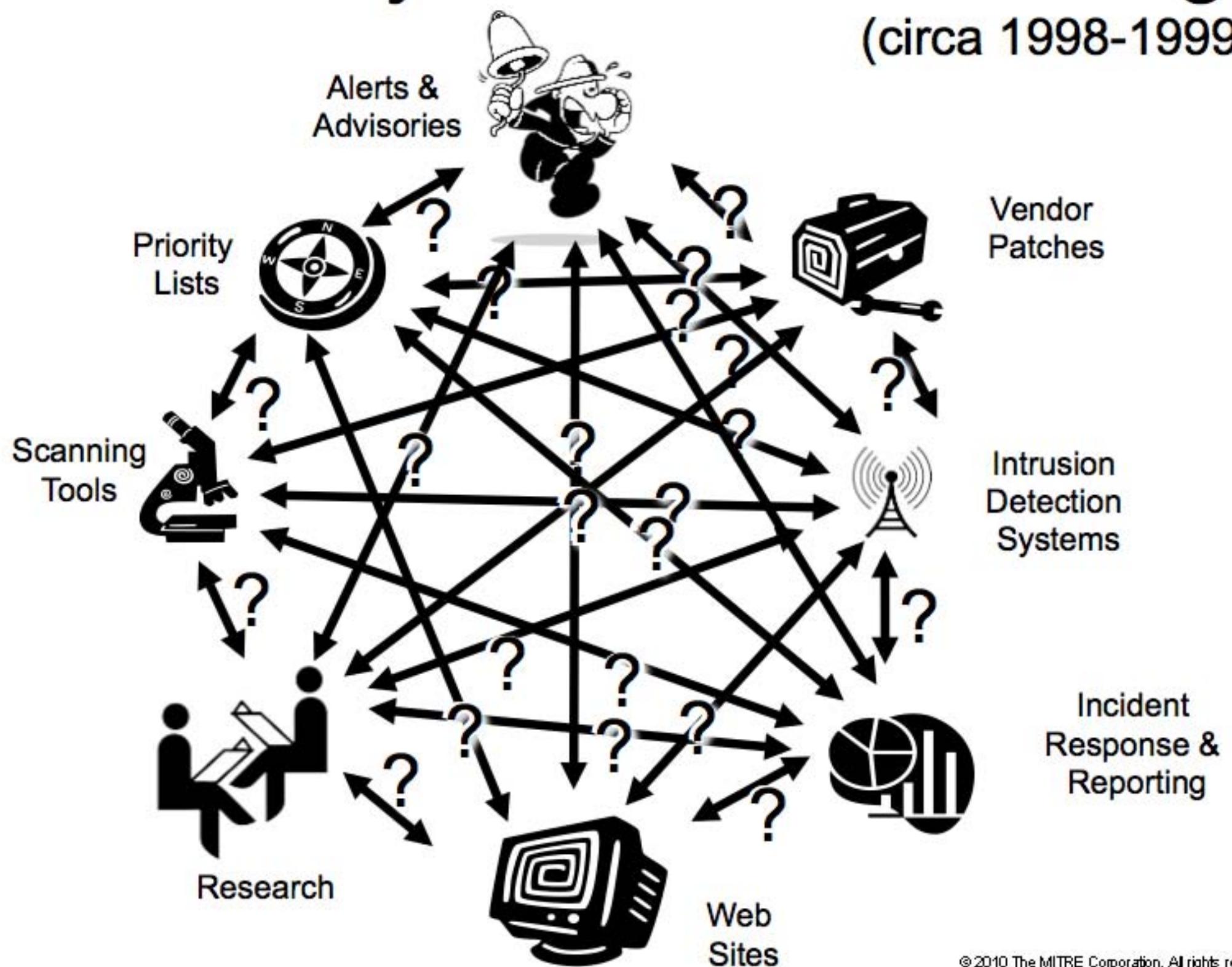
- Enumerations
  - Catalog the fundamental entities in IA, Cyber Security, and Software Assurance
    - Vulnerabilities (CVE), configuration issues (CCE), software packages (CPE), attack patterns (CAPEC), weaknesses in code/design/architecture (CWE)
- Languages/Formats
  - Support the creation of machine-readable state assertions, assessment results, and messages
    - Configuration/vulnerability/patch/asset patterns (XCCDF & OVAL), results from standards-based assessments (ARF), software security patterns (SBVR), event patterns (CEE), malware patterns (MAEC), risk of a vulnerability (CVSS), config risk (CCSS), weakness risk (CWSS), information messages (CAIF & \*DEF)
- Knowledge Repositories
  - Packages of assertions supporting a specific application
    - Vulnerability advisories & alerts, (US-CERT Advisories/IAVAs), configuration assessment (NIST Checklists, CIS Benchmarks, NSA Configuration Guides, DISA STIGS), asset inventory (NIST/DHS NVD), code assessment & certification (NIST SAMATE, DoD DIACAP & eMASS)

## Tools

- Interpret IA, Cyber Security, and SwA content in context of enterprise network
- Methods for assessing compliance to languages, formats, and enumerations

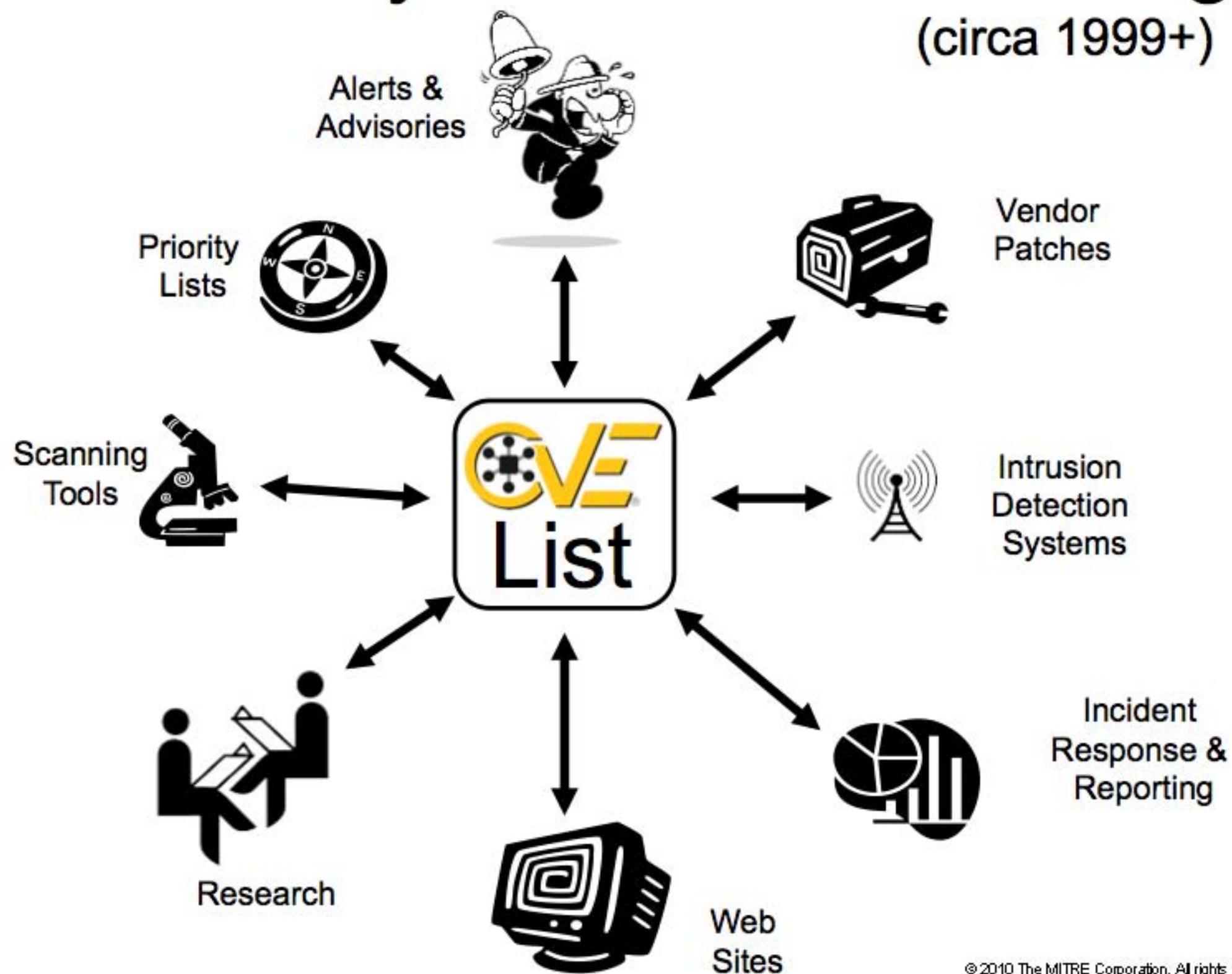
# Vulnerability Information Sharing

(circa 1998-1999)



# Vulnerability Information Sharing

(circa 1999+)



Click Here to Install Silverlight

United States Change | All Microsoft Sites

[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 impacted than users who operate with administrative user rights.

[↑ 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](#)

Run Quick Links

[Sign In/Register for Account](#) | [Help](#)

United States

Communities

I am a...

I want to...

 Secure Search

Products and Services

Downloads

Store

Support

Education

Partners

About

Oracle Technology Network

Oracle Technology Network &gt; Topics &gt; Security

Embedded

BI &amp; Data Warehousing

.NET

Linux

PHP

## Oracle Critical Patch Update Advisory - October 2010

### Description

A Critical Patch Update is a collection of patches for multiple security vulnerabilities. It also includes non-security fixes that are required (because of interdependencies) by those security patches. Critical Patch Updates are cumulative, except as noted below, but each advisory describes only the security fixes added since the previous Critical Patch Update. Thus, prior Critical Patch Update Advisories should be reviewed for information regarding earlier accumulated security fixes. Please refer to:

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)	JM 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	OLTP	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	DK	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	Oracle 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

[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 \(v. 5 client\)](#)

OVAL: [com:redhat\\_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](#)



Store

Mac

iPod

iPhone

iPad

iTunes

Support

Search

Mailing Lists

# Apple Mailing Lists

  Search only in security-announce list[\[Date Prev\]](#) [\[Date Next\]](#) [\[Thread Prev\]](#) [\[Thread Next\]](#) [\[Date Index\]](#) [\[Thread Index\]](#)

## 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](mailto:email@hidden)>**Date:** Wed, 11 Aug 2010 12:19:43 -0700**Delivered-to:** [email@hidden](mailto:email@hidden)**Delivered-to:** [email@hidden](mailto: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

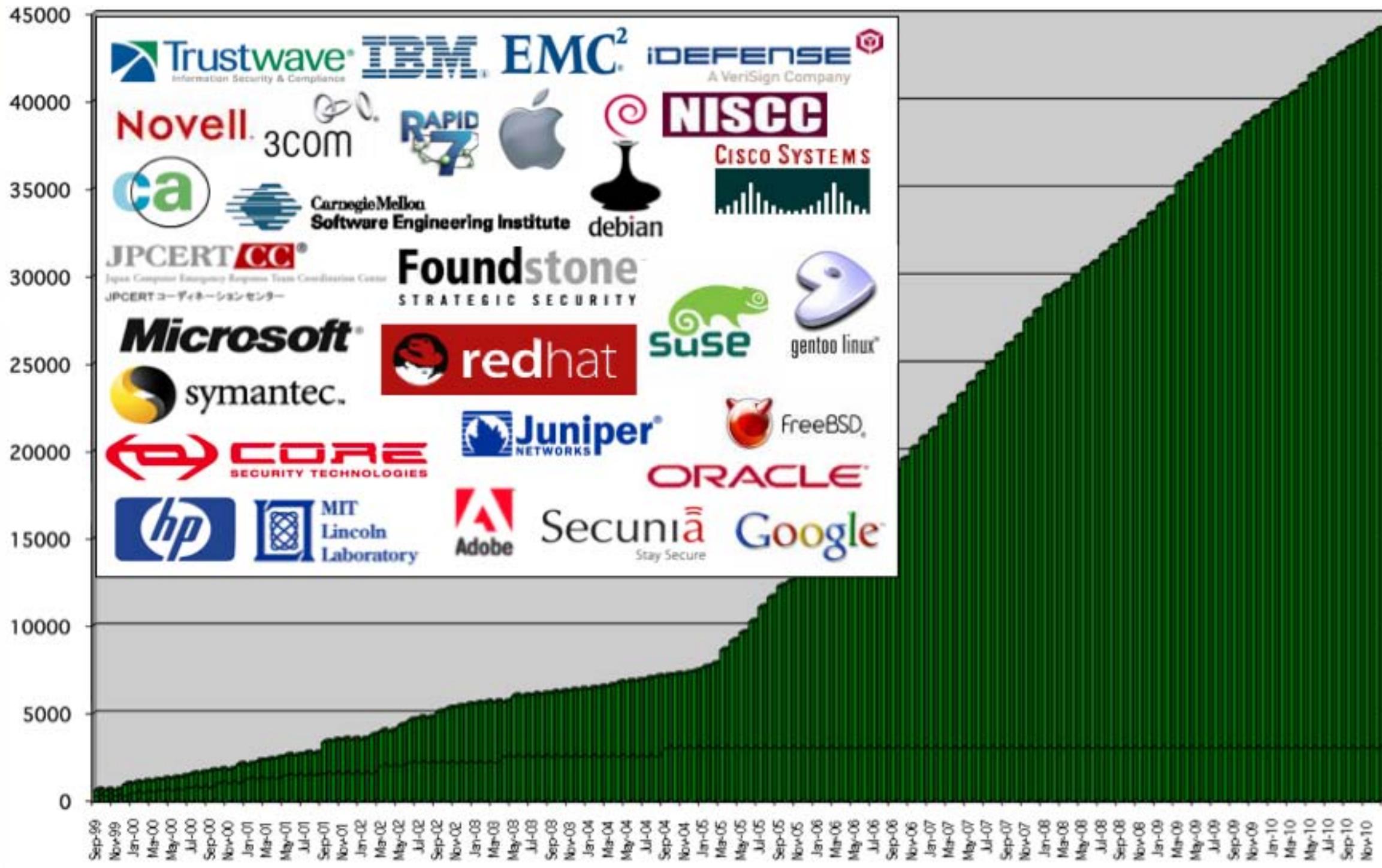
CVE-ID: CVE-2010-1797

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 executionDescription: A stack buffer overflow exists in FreeType's handling  
of CFF encodes. Viewing a PDF document with maliciously crafted

# CVE 1999 to 2010



# CVE is Widely Used & Available 44,144 and climbing...

The image displays a grid of 24 screenshots of the CVE homepage, each in a different language. The languages shown are: Arabic, Bulgarian, Catalan, Chinese, Croatian, Czech, Danish, Dutch, Estonian, Finnish, French, German, Hebrew, Hungarian, Icelandic, Indonesian, Italian, Japanese, Korean, Latvian, Lithuanian, Norwegian, Polish, Portuguese, Romanian, Russian, Serbian, Slovak, Slovenian, Spanish, Swedish, and Turkish. The screenshots illustrate the international reach and availability of the Common Vulnerabilities and Exposures (CVE) database.

**Arabic**

**Bulgarian**

**Catalan**

**Chinese**

**Croatian**

**Czech**

**Danish**

**Dutch**

**Estonian**

**Finnish**

**French**

**German**

**Hebrew**

**Hungarian**

**Icelandic**

**Indonesian**

**Italian**

**Japanese**

**Korean**

**Latvian**

**Lithuanian**

**Norwegian**

**Polish**

**Portuguese**

**Romanian**

**Russian**

**Serbian**

**Slovak**

**Slovenian**

**Spanish**

**Swedish**

**Turkish**

**Common Vulnerabilities and Exposures (CVE)**

**About CVE**

**Widespread Adoption of CVE**

- Vulnerability Management
- Patch Management
- Vulnerability Alerting
- Intrusion Detection
- NVD (National Vulnerability Database)
- US-CERT Bulletin
- SANS Top 20

**Similar Standards**

Common漏洞 (CNVD)  
漏洞特征码 (CNVD)  
漏洞特征 (CNVD)  
漏洞报告 (CNVD)

Common Vulnerabilities and Exposures (CVE)  
Assessment Language (CALM)  
Security Content Automation (SCAP)  
Metadata Repository (MDR)

**Focus On**

CVE Identifiers

CVE Identifiers (also called "CVE IDs," "CVE names," "CVE numbers," and "CVNs") are unique, common identifiers for publicly known information security vulnerabilities. Each CVE Identifier on the CVE List includes a CVE identifier number (e.g., "CVE-1999-0001"); indication of "entry" or "candidate" status; a brief description of the security vulnerability or exposure; and any pertinent references (e.g., vulnerability reports and advisories or NIST IRs). CVE Identifiers are used by information security products/services/vendors and researchers as a standard method for identifying vulnerabilities and for cross-linking with other resources that also use CVE Identifiers.

Page last updated: August 26, 2009

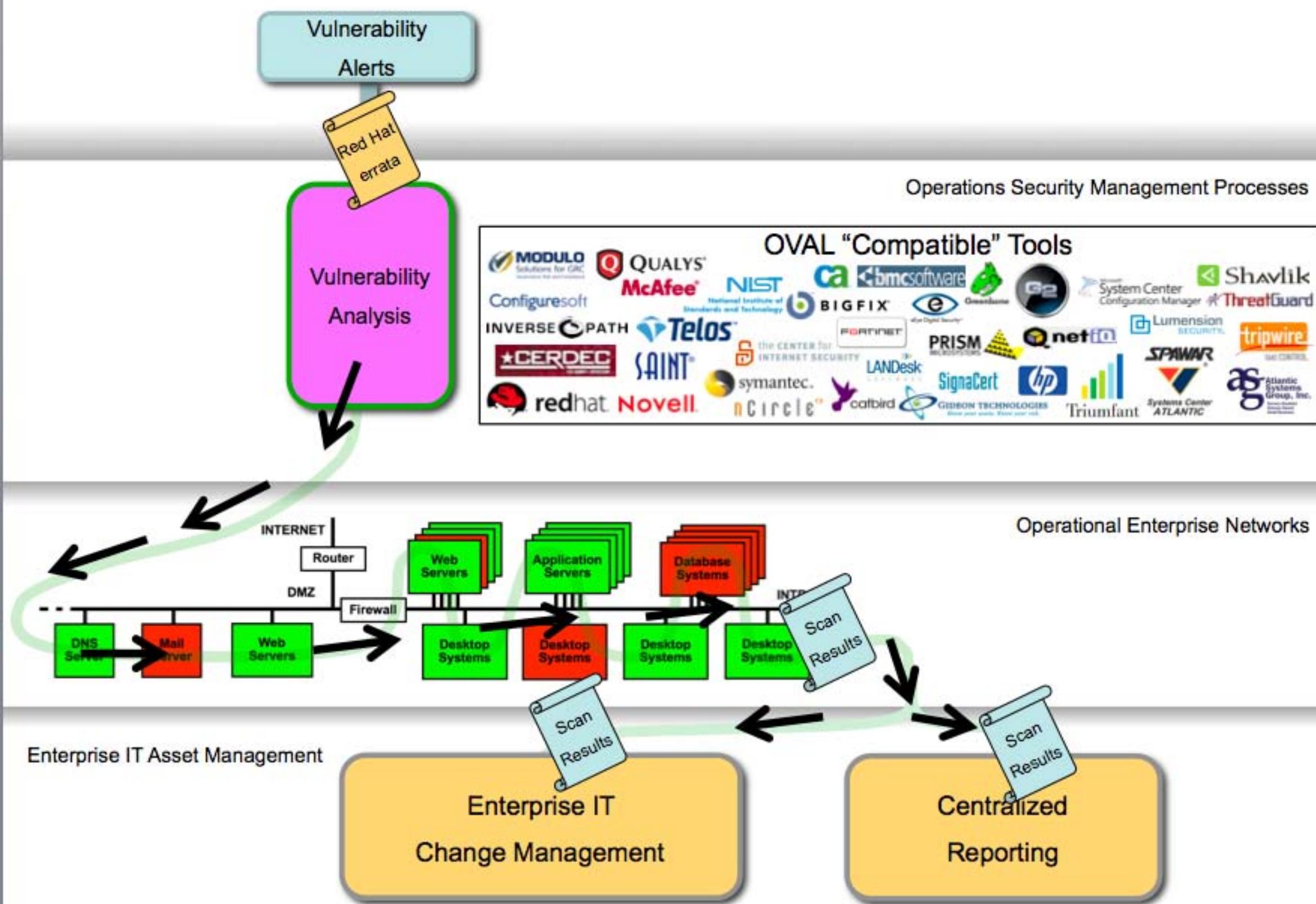
Use of the Common Vulnerabilities and Exposures (CVE) and its associated reference data does not imply endorsement. The use of this site is subject to the Terms of Use. For more information, please visit [cve.mitre.org](#).

CVE is operated by the Defense Cyber Infrastructure Division of the U.S. Department of Homeland Security (DHS), Stop Exploitation. DHS and the CVE logo are trademarks of the DHS. The DHS logo is a trademark of the United States Government and is used under license. This work was created by DHS. © 2009 DHS. All rights reserved.

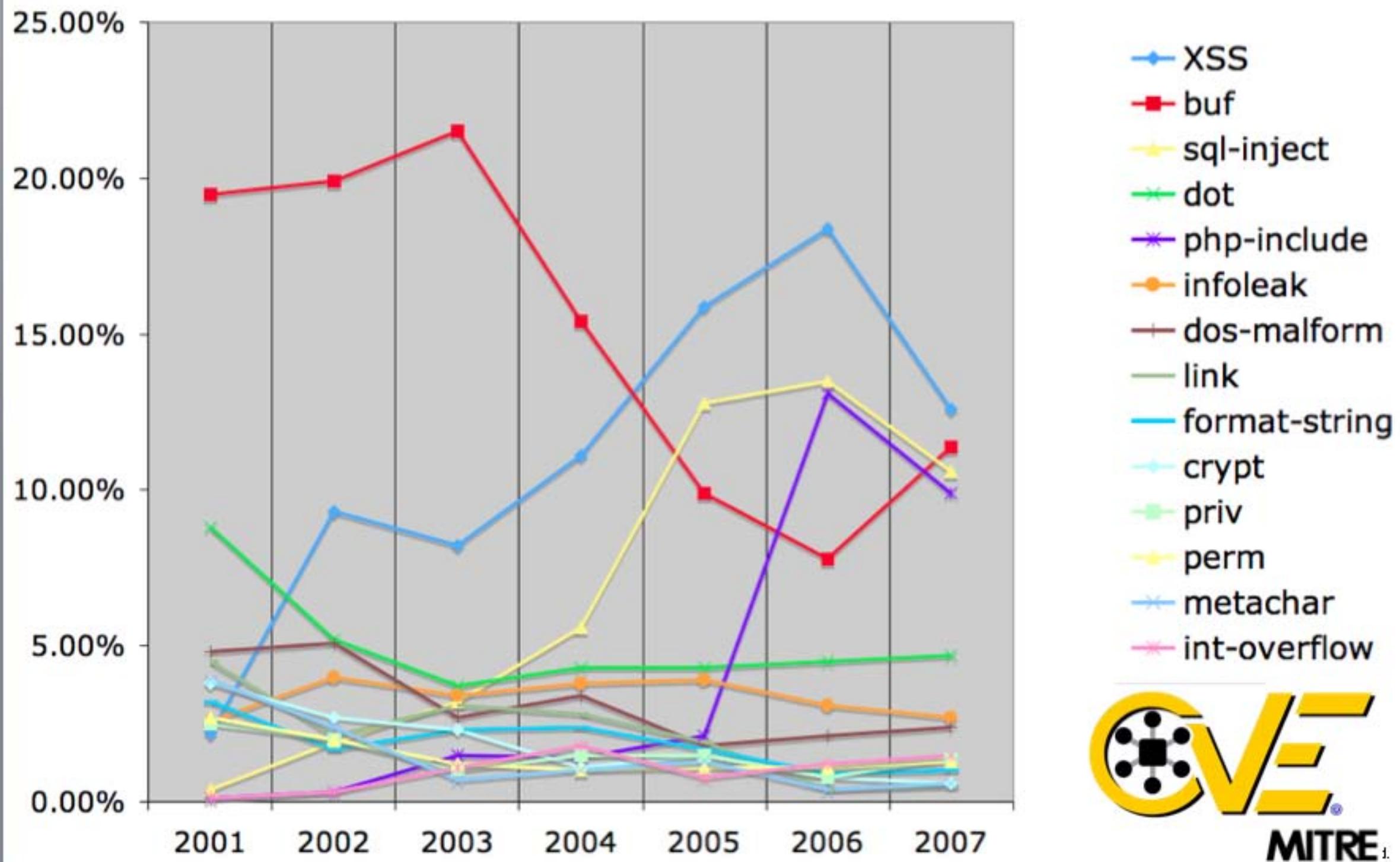
Privacy policy  
Terms of Use  
Contact us

# CVE Vendor/Industry Engagement





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



# Removing and Preventing the Vulnerabilities Requires More Specific Definitions...CWEs

XSS

buf

sql-inject

dot

php-include

infoleak

dos-malform

link

format-string

crypt

priv

perm

metachar

int-overflow

Failure to Sanitize Directives in a Web Page (aka 'Cross-site scripting' (XSS)) (79)

- Failure to Sanitize Script-Related HTML Tags in a Web Page (Basic XSS) (80)
- Failure to Sanitize Directives in an Error Message Web Page (81)
- Failure to Sanitize Script in Attributes of IMG Tags in a Web Page (82)
- Failure to Sanitize Script in Attributes in a Web Page (83)
- Failure to Resolve Encoded URI Schemes in a Web Page (84)
- Doubled Character XSS Manipulations (85)
- Invalid Characters in Identifiers (86)
- Alternate XSS syntax (87)

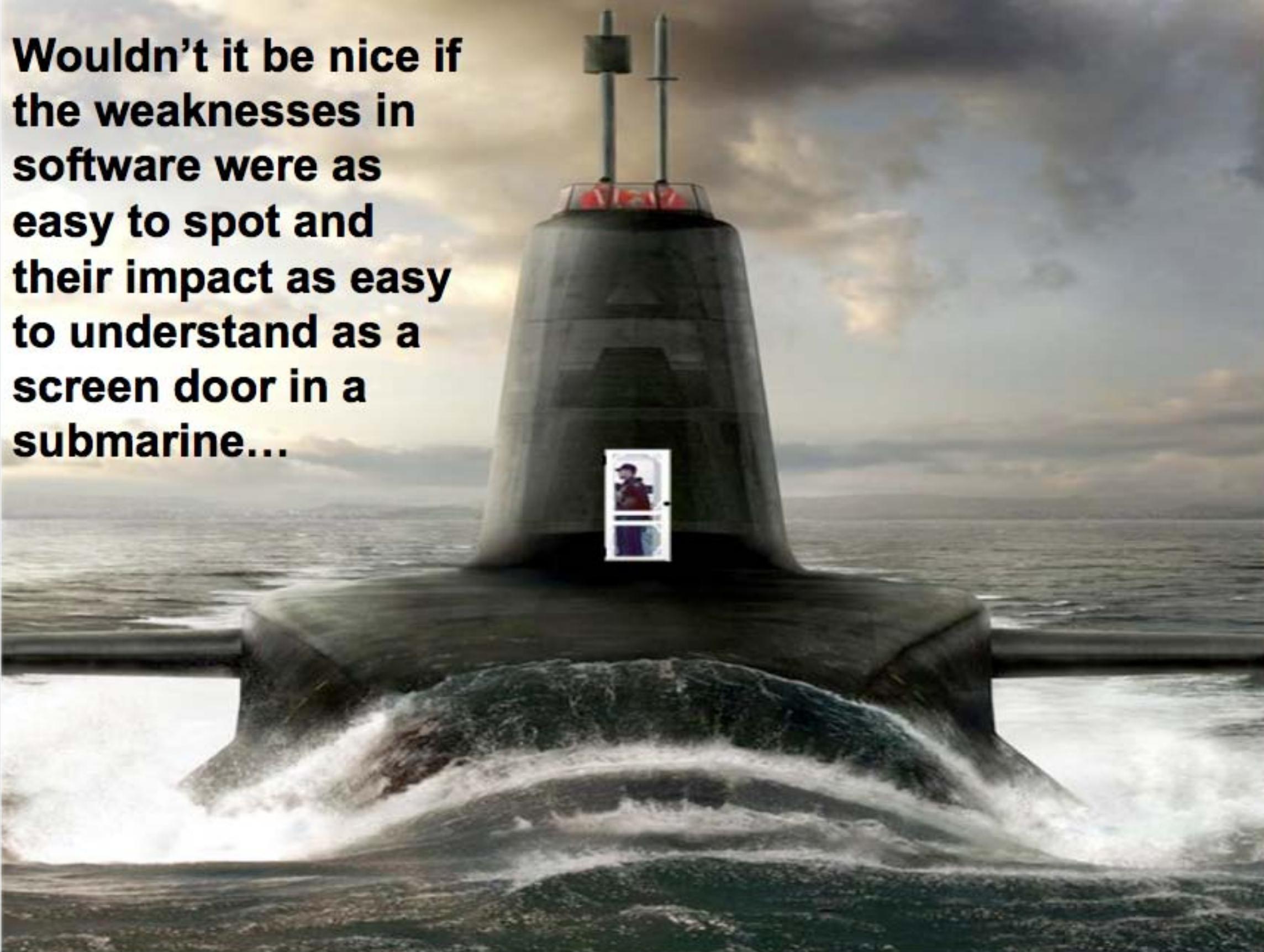
Failure to Constrain Operations within the Bounds of an Allocated Memory Buffer (119)

- Unbounded Transfer ('Classic Buffer Overflow') (120)
- Write-what-where Condition (123)
- Boundary Beginning Violation ('Buffer Underwrite') (124)
- Out-of-bounds Read (125)
- Wrap-around Error (128)
- Unchecked Array Indexing (129)
- Incorrect Calculation of Buffer Size (131)
- Miscalculated Null Termination (132)
- Return of Pointer Value Outside of Expected Range (466)

Path Traversal (22)

- Relative Path Traversal (23)
  - Path Traversal: '..\filename' (29)
  - Path Traversal: 'dir..\filename' (30)
  - Path Traversal: 'dir..\filename' (31)
  - Path Traversal: '...' (Triple Dot) (32)
  - Path Traversal: '....' (Multiple Dot) (33)
  - 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)

**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...**

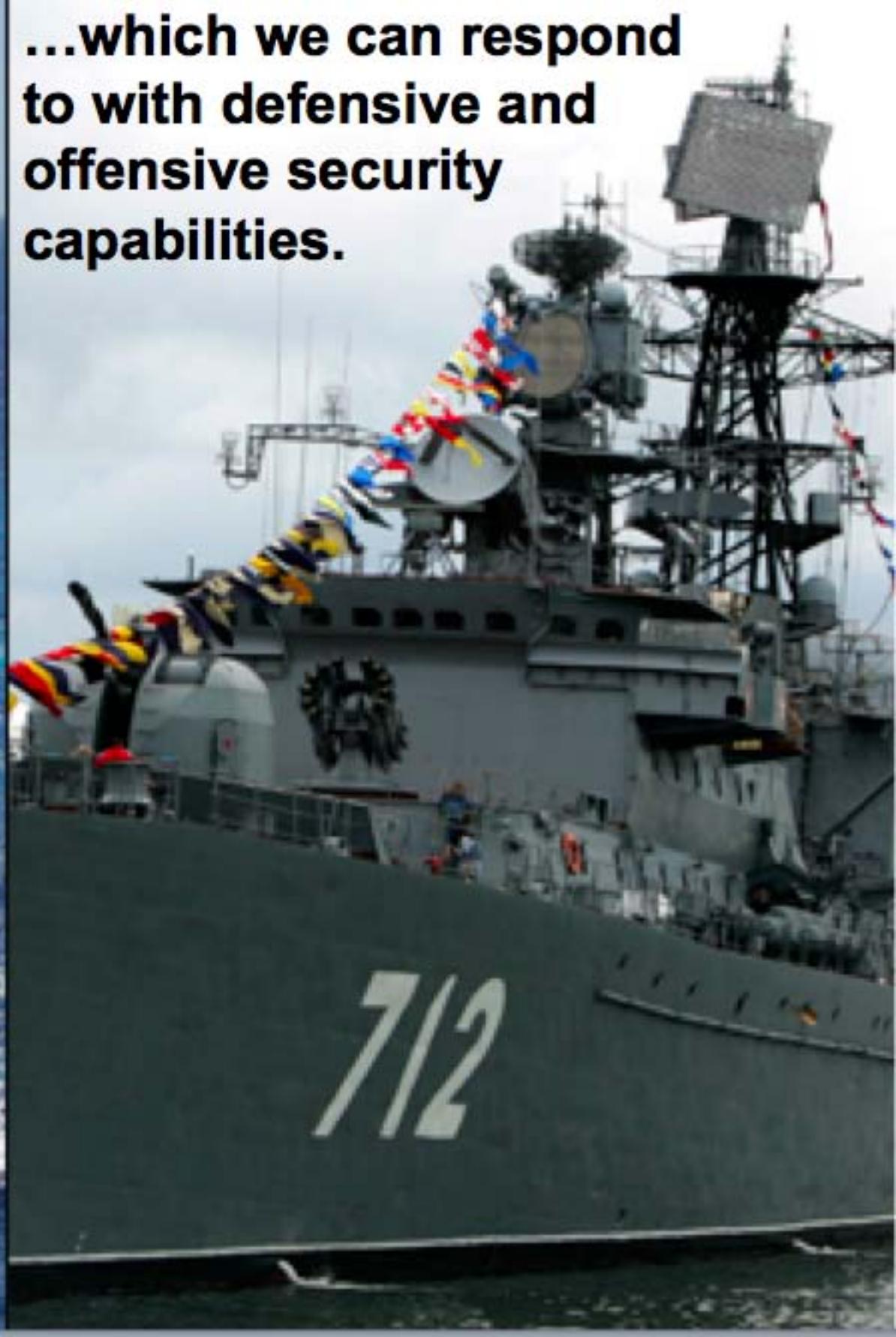


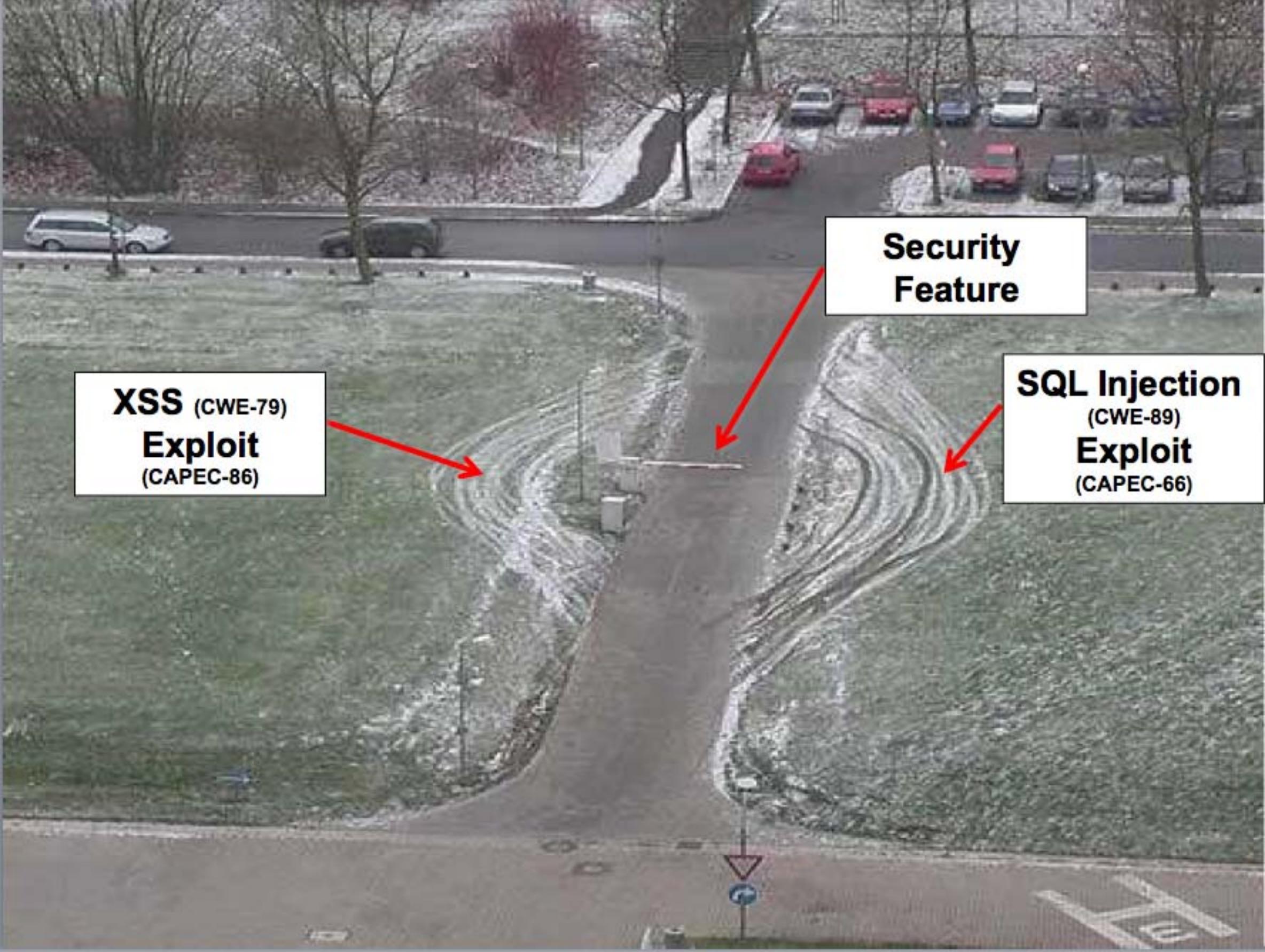
**But remember, there are people that are out there trying to take advantage of vulnerabilities and weaknesses in your technologies, processes, or practices...**





**...which we can respond to with defensive and offensive security capabilities.**





**XSS (CWE-79)**  
**Exploit**  
(CAPEC-86)

**Security**  
**Feature**

**SQL Injection**  
(CWE-89)  
**Exploit**  
(CAPEC-66)



# Common Weakness Enumeration

A Community-Developed Dictionary of Software Weakness Types



Home &gt; CWE/SANS Top 25 2010

Search by ID:  Go**CWE List**
[Full Dictionary View](#)  
[Development View](#)  
[Research View](#)  
[Reports](#)
**About**
[Sources](#)  
[Process](#)  
[Documents](#)
**Community**
[Related Activities](#)  
[Discussion List](#)  
[Research](#)  
[CWE/SANS Top 25](#)  
[CWSS](#)
**News**
[Calendar](#)  
[Free Newsletter](#)
**Compatibility**
[Program Requirements](#)  
[Declarations](#)  
[Make a Declaration](#)
**Contact Us**
[Search the Site](#)

## 2010 CWE/SANS Top 25 Most Dangerous Software Errors

Copyright © 2010

<http://cwe.mitre.org/top25/>

The MITRE Corporation

**Document version:** 1.06 ([pdf](#))**Date:** September 27, 2010**Project Coordinators:**
 Bob Martin (MITRE)  
 Mason Brown (SANS)  
 Alan Paller (SANS)  
 Dennis Kirby (SANS)
**Document Editor:**

Steve Christey (MITRE)

**Section Contents****CWE/SANS Top 25**
[Contributors](#)  
[Supporting Quotes](#)  
[Monster Mitigations](#)  
[Focus Profiles](#)  
[On the Cusp](#)  
[Documents & Podcasts](#)  
[Training Materials](#)  
[Top 25 FAQ](#)  
[Top 25 Process](#)  
[Change Log](#)
**SANS News Release****Section Archives**
[2009 CWE/SANS Top 25](#)  
[Supporting Quotes](#)  
[Contributors](#)  
[On The Cusp](#)  
[Change Log](#)

### Introduction

The 2010 CWE/SANS Top 25 Most Dangerous Software Errors is a list of the most widespread and critical programming errors that can lead to serious software vulnerabilities. They are often easy to find, and easy to exploit. They are dangerous because they will frequently allow attackers to completely take over the software, steal data, or prevent the software from working at all.

The Top 25 list is a tool for education and awareness to help programmers to prevent the kinds of vulnerabilities that plague the software industry, by identifying and avoiding all-too-common mistakes that occur before software is even shipped. Software customers can use the same list to help them to ask for more secure software. Researchers in software security can use the Top 25 to



## 2

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

### Summary

Weakness Prevalence	High	Consequences	Data loss, Security bypass
Remediation Cost	Low	Ease of Detection	Easy
Attack Frequency	Often	Attacker Awareness	High

### Discussion

These days, it seems as if software is all about the data: getting it into the database, pulling it from the database, massaging it into information, and sending it elsewhere for fun and profit. If attackers can influence the SQL that you use to communicate with your database, then suddenly all your fun and profit belongs to them. If you use SQL queries in security controls such as authentication, attackers could alter the logic of those queries to bypass security. They could modify the queries to steal, corrupt, or otherwise change your underlying data. They'll even steal data one byte at a time if they have to, and they have the patience and know-how to do so.

[Technical Details](#) | [Code Examples](#) | [Detection Methods](#) | [References](#)

### Prevention and Mitigations

#### Architecture and Design

Use a vetted library or framework that does not allow this weakness to occur or provides constructs that make this weakness easier to avoid.

For example, consider using persistence layers such as Hibernate or Enterprise Java Beans, which can provide significant protection against SQL injection if used properly.

#### Architecture and Design

If available, use structured mechanisms that automatically enforce the separation between data and code. These mechanisms may be able to provide the relevant quoting, encoding, and validation automatically, instead of relying on the developer to provide this capability at every point where output is generated.

Process SQL queries using prepared statements, parameterized queries, or stored procedures. These features should accept parameters or variables and support strong typing. Do not dynamically construct and execute query strings within these features using "exec" or similar functionality, since you may

# CAPEC Common Attack Pattern Enumeration and Classification

A Community Knowledge Resource for Building Secure Software

Home &gt; CAPEC List &gt; CAPEC-66: SQL Injection (Release 1.5)

Search by ID:  Go**CAPEC List**
[Full CAPEC Dictionary](#)  
[Methods of Attack View](#)  
[Reports](#)
**About CAPEC**
[Documents](#)  
[Resources](#)
**Community**
[Related Activities](#)  
[Collaboration List](#)  
[News & Events](#)  
[Calendar](#)  
[Free Newsletter](#)
**Contact Us**[Search the Site](#)

## CAPEC-66: SQL Injection

### SQL Injection

**Attack Pattern ID:** 66 ([Standard](#))  
**Attack Pattern Completeness:** [Complete](#)
**Typical Severity:** High**Status:** Draft

CAPEC - CAPEC-7: Blind SQL Injection (Release 1.5)

**Description****Summary**

This attack exploits target so An attacker crafts input string statements based on the input those the application intends SQL Injection results from specially crafted user-controlled validation as part of SQL query ways not envisaged during a design of the application, it enables a database execute system-re enables an attacker to talk completely. Successful injection or modify data in the database information from a database

**Attack Execution Flow****Explore****1. Survey application:**

The attacker first takes an in

**Attack Step Techniques****ID** **Attack Step Technique**

- | ID | Attack Step Technique    |
|----|--------------------------|
| 1  | Spider web sites for all |
| -  | -                        |

Done

### SQL Injection

Attack Pattern ID: 66 ([Standard](#)) Attack Pattern Completeness: [Complete](#) Typical Severity: High Status: Draft CAPEC - CAPEC-7: Blind SQL Injection (Release 1.5) Search by ID: Go

## CAPEC-7: Blind SQL Injection

### Blind SQL Injection

Attack Pattern ID: 7 ([Detailed](#) Attack Pattern) Completeness: [Complete](#) Typical Severity: High Status: Draft

#### Description

#### Summary

Blind SQL Injection results from an insufficient mitigation for SQL Injection. Although suppressing database error messages are considered best practice, the suppression alone is not sufficient to prevent SQL Injection. Blind SQL Injection is a form of SQL Injection that overcomes the lack of error messages. Without the error messages that facilitate SQL Injection, the attacker constructs input strings that probe the target through simple Boolean SQL expressions. The attacker can determine if the syntax and structure of the injection was successful based on whether the query was executed or not. Applied iteratively, the attacker determines how and where the target is vulnerable to SQL Injection.

For example, an attacker may try entering something like "username' AND 1=1; --" in an input field. If the result is the same as when the attacker entered "username" in the field, then the attacker knows that the application is vulnerable to SQL Injection. The attacker can then ask yes/no questions from the database server to extract information from it. For example, the attacker can extract table names from a database using the following types of queries:

```
"username' AND ascii(lower(substring((SELECT TOP 1 name FROM sysobjects WHERE xtype='U'), 1, 1))) > 108".
If the above query executes properly, then the attacker knows that the first character in a table name in the database is a letter between m and z. If it doesn't, then the attacker knows that the character must be between a and i (assuming of course that table names only contain alphabetic characters). By performing a binary search on all character positions, the attacker can determine all table names in the database. Subsequently, the attacker may execute an actual attack and send something like:
"username'; DROP TABLE trades; --"
```

#### Attack Execution Flow

#### Explore

#### 1. Hypothesize SQL queries in application:

Generated hypotheses regarding the SQL queries in an application. For example, the attacker may hypothesize that his input is passed directly into a query that looks like:

```
"SELECT * FROM orders WHERE ordernum = _____"
or
"SELECT * FROM orders WHERE ordernum IN (_____)"
```

Done



## Severity



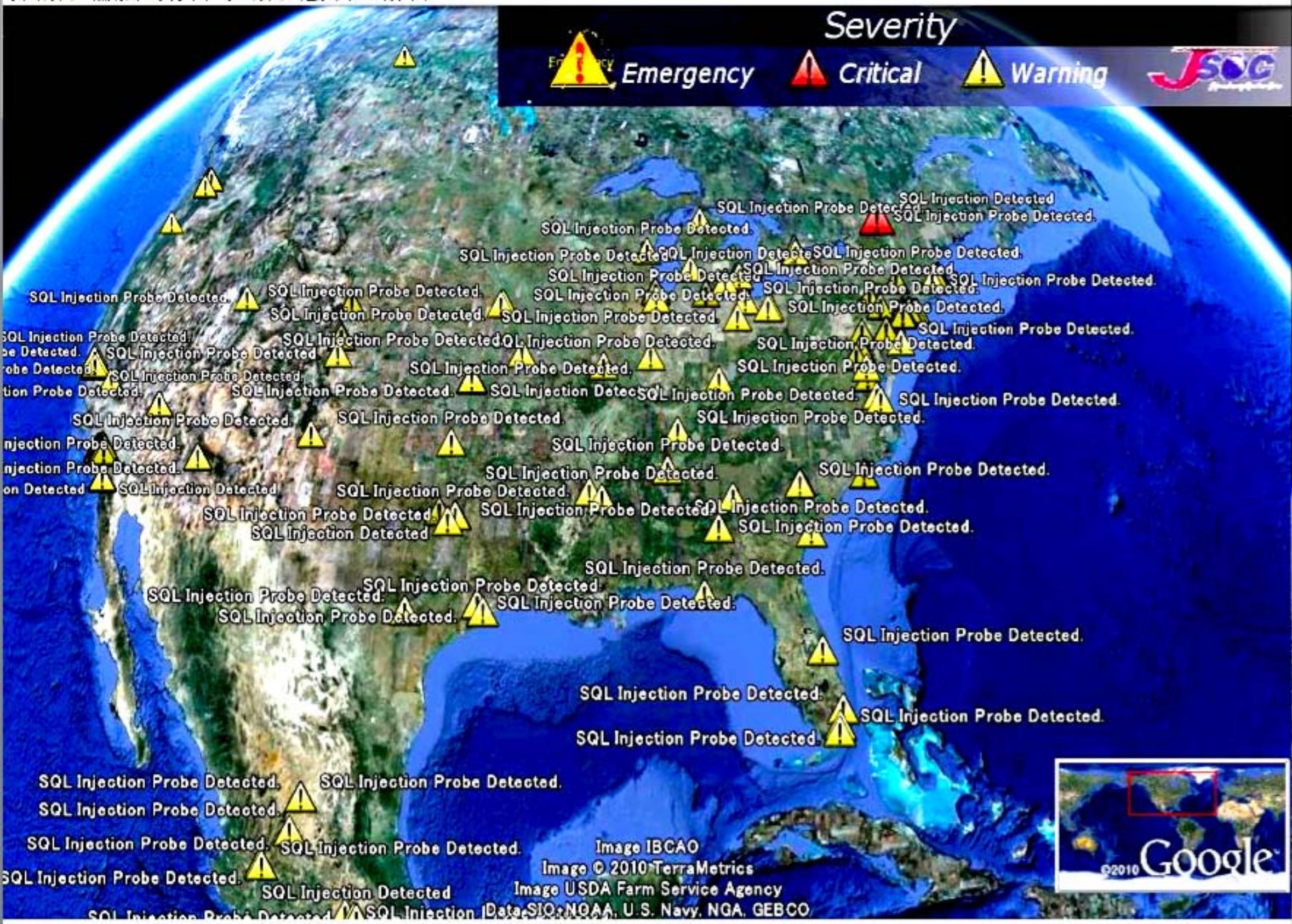
Emergency



Critical



Warning



# CWE Compatibility & Effectiveness Program

(launched Feb 2007)

CWE - CWE Compatibility

http://cwe.mitre.org/compatible/index.html

AFC Home MIT Home Search Map/Ph/Weather/Travel Bob's Bookmarks CVEinOVAL OVAL shared SPAMmngt LogosofSPAMmngt

**CWE Common Weakness Enumeration**  
A community-developed dictionary of common software weaknesses

Home > Compatibility

CWE List Full Dictionary View Change Request Form

**CWE Compatibility**

View the CWE List

Section Contents  
Compatibility



## Organizations Participating

All organizations participating in the CWE Compatibility and Effectiveness Program are listed below, including those with CWE-Compatible Products and Services and those with Declarations to Be CWE-Compatible.

Products are listed alphabetically by organization name:

cwe.mitre.org/compatible/

### TOTALS

Organizations Participating: 29  
Products & Services: 48

December 29, 2006



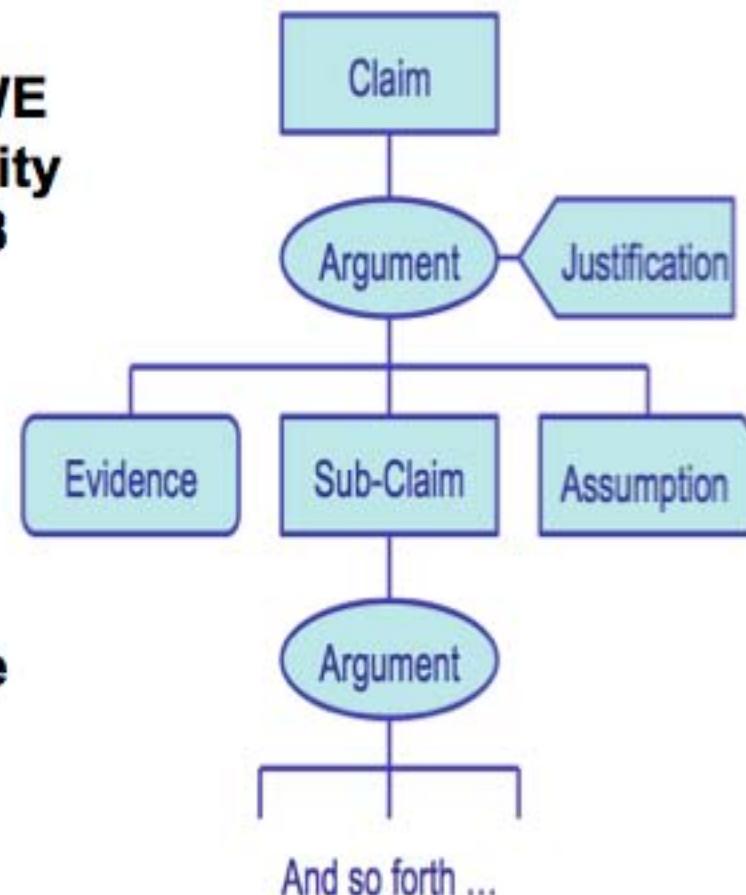
<b>ISO IEC</b>	ISO/IEC JTC 1/SC 27 Nxxxxx
	ISO/IEC JTC 1/SC 27/WG x Nxxxxxx
	REPLACES N
	ISO/IEC JTC 1/SC 27 Information technology - Security techniques Secretariat: DIN, Germany
DOC TYPE:	ISO/HWI Proposal for a technical report (TR)
TITLE:	National Body New Work Item Proposal on "Secure software development and evaluation under ISO/IEC 15408 and ISO/IEC 18045"
SOURCE:	MC/ITB/1/01, National Body of (DE)
DATE:	2009-09-09
PROJECT:	18408 and 18405
STATUS:	This document is circulated for consideration at the forthcoming meeting of SC 27/WG 0. It is to be held in Rydevald (NVA, USA) on 2 <sup>nd</sup> - 6 <sup>th</sup> November 2009.
ACTION ID:	ACT
DUE DATE:	
DISTRIBUTION:	P. O. and L. Mombour W. Furtig, SC 27 Chairman M. De Ruwe, SC 27 Vice-Chair S. J. Humphreys, K. Nomura, M. Rehm, H.-C. Kang, K. Rammertzberg, MC-Committee
MEDIUM:	Linux-server
NO. OF PAGES:	01

## Common Criteria v4 CCDB

- TOE to leverage CAPEC & CWE
- “Refining Software Vulnerability Analysis Under ISO/IEC 15408 and ISO/IEC 18045”
- Also investigating how to leverage ISO/IEC 15026 and OMG’s Structured Assurance Case Metamodel

## NIAP (U.S.) Evaluation Scheme

- Above plus
- Also investigating how to leverage SCAP



## Knowledge Repositories

Attack Pattern Knowledge

Security Weakness Knowledge

Malware Knowledge

Asset Knowledge

Vulnerability Knowledge

Configuration Knowledge



System & Software Assurance Guidance/ Requirements  
For Developed Items

System & Software Assurance Guidance/ Requirements  
For Purchased Items

CWE/CAPEC/  
SBVR/MAEC

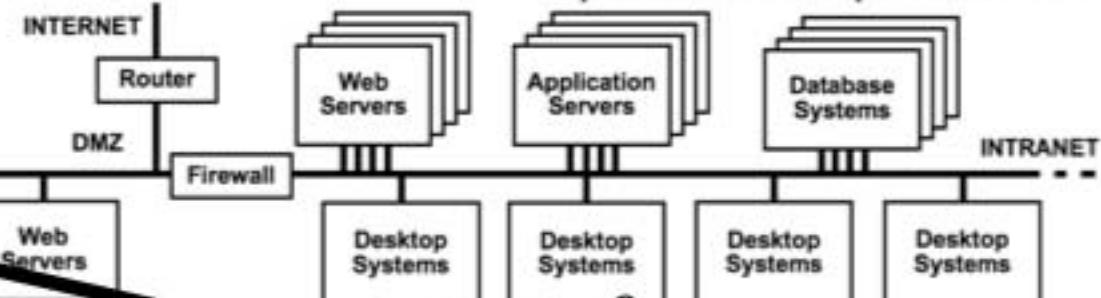
SVL/XCCDF/  
CCE/CPE/ARF

System & Software  
Assurance Assessment

Assessment of  
System  
Development,  
Integration, &  
Sustainment  
Activities and  
Certification &  
Accreditation

Development & Sustainment Security Management Processes

Operational Enterprise Networks

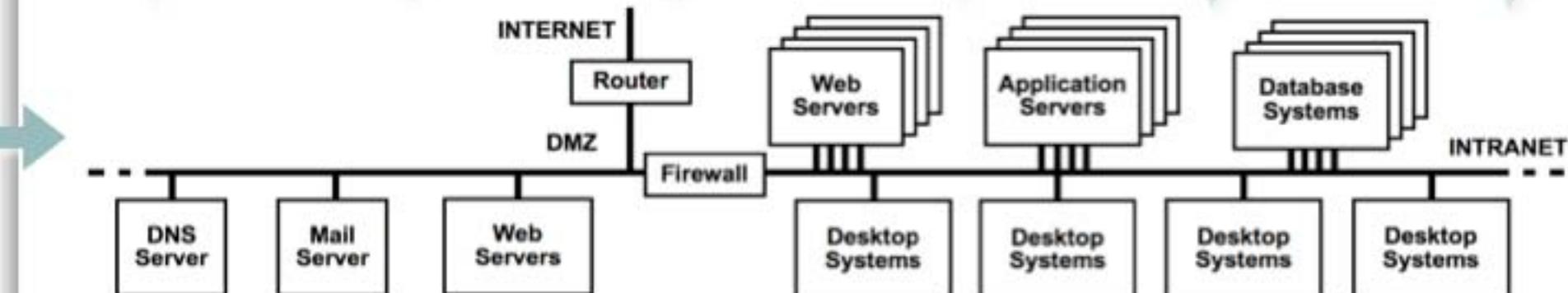
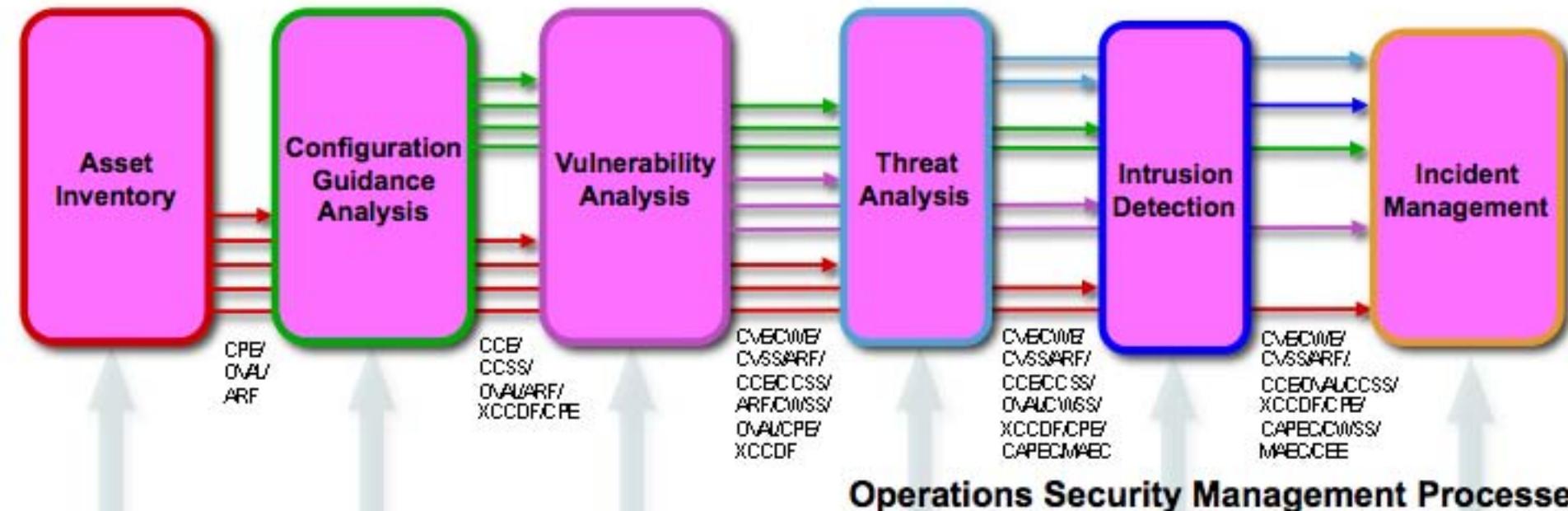


Enterprise IT Asset Management

Enterprise IT  
Change Management

Centralized Reporting





**Operational Enterprise Networks**

**Development & Sustainment Security Management Processes**

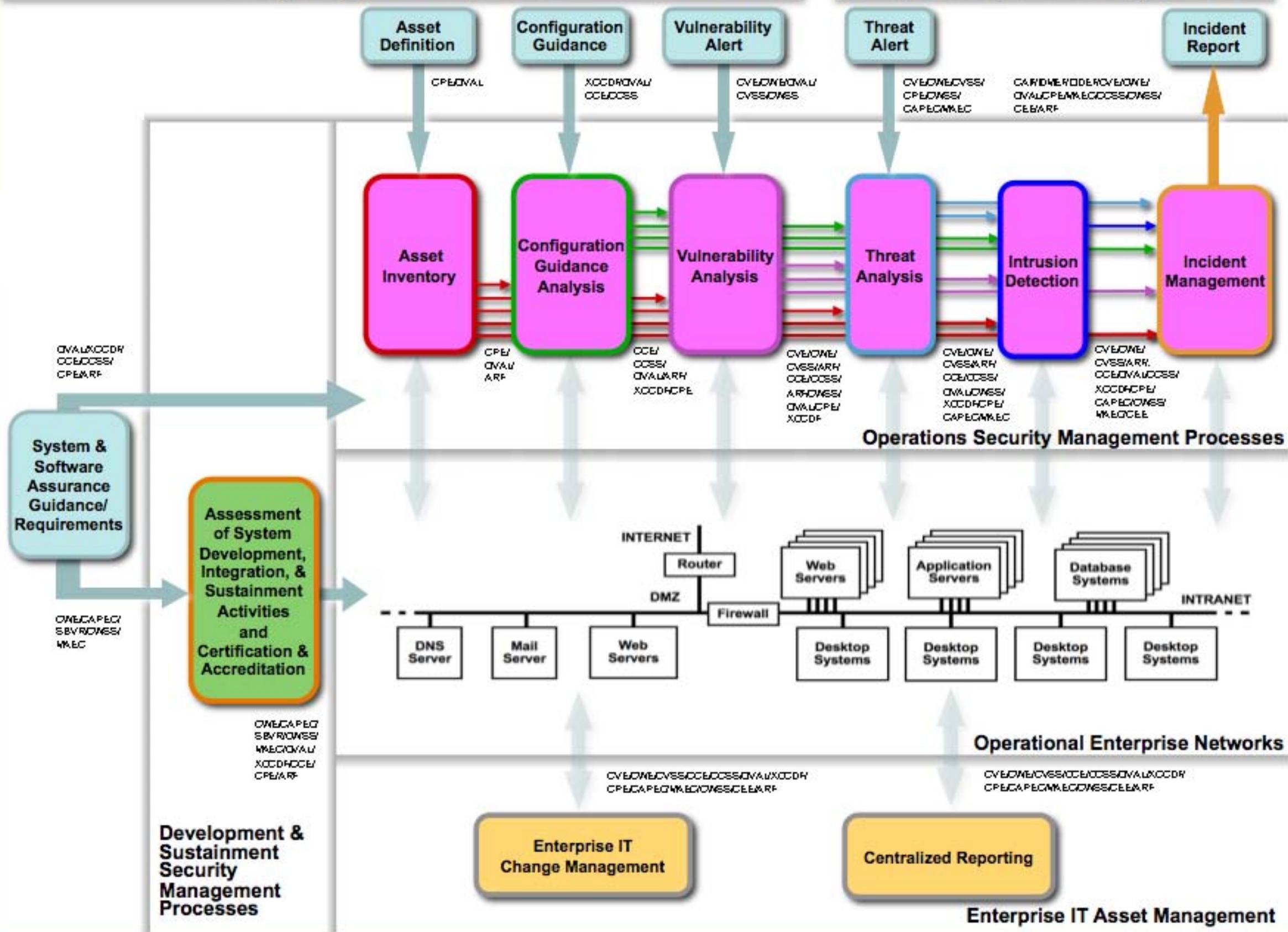
Enterprise IT Change Management

Centralized Reporting

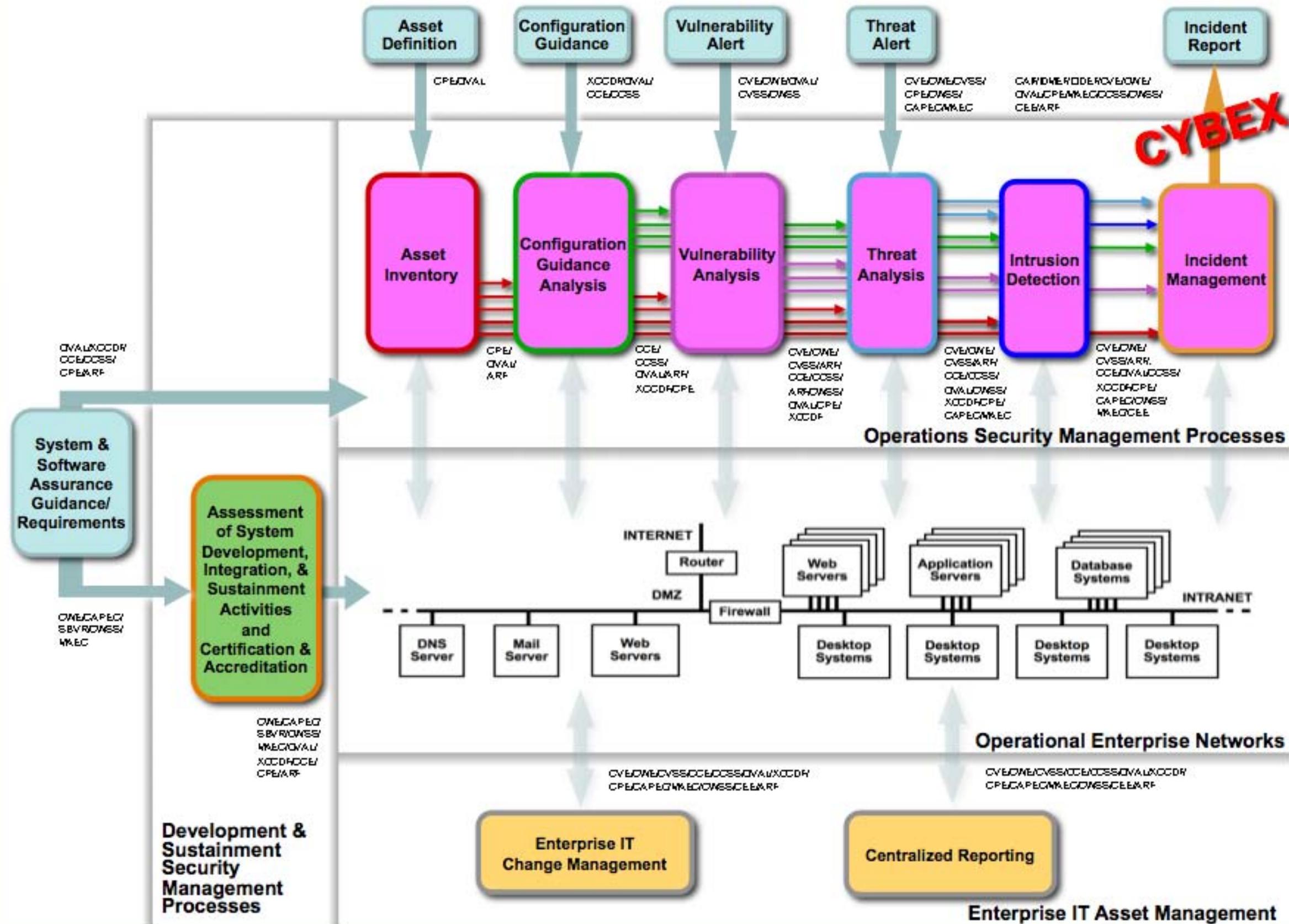
**Enterprise IT Asset Management**

## Mitigating Risk Exposures

## Responding to Security Threats



# Knowledge Repositories



# [makingsecuritymeasurable.mitre.org]

Making Security Measurable

http://msm.mitre.org/ Google

MITRE, in collaboration with government, industry, and academic stakeholders, is improving the measurability of security through enumerating baseline security data, providing standardized languages as means for accurately communicating the information, and encouraging the sharing of the information with users by developing repositories.

The other activities and initiatives listed here have similar concepts or compatible approaches to MITRE's. Together all of these efforts are helping to make security more measurable by defining the concepts that need to be measured, providing for high fidelity communications about the measurements, and providing for sharing of the measurements and the definitions of what to measure.

Done

Making Security Measurable

A Collection of Information Security Community Standardization Activities and Initiatives

Home | About | Current Initiatives | Initiator | Events & Participants | Feedback Requested

Measurable security pertains at a minimum to the following areas:

- Vulnerability Management
- Intrusion Detection
- Asset Security Assessment
- Asset Management
- Configuration Guidance
- Patch Management
- Malware Response
- Incident Management
- Threat Analysis

Enumerations	Languages	Repositories
Common Vulnerabilities and Exposures (CVE™) - common vulnerability identifiers	Open Vulnerability and Assessment Language (OVAL™) - standard for determining vulnerability and configuration issues	OVAL Repository - community-developed OVAL Vulnerability, Compliance, Inventory, and Patch Definitions
Common Weakness Enumeration (CWE™) - list of software weakness types	Common Event Expression (CEE™) - standardizes the way computer events are described, logged, and exchanged	National Vulnerability Database (NVD) - U.S. vulnerability database based on CVE that integrates all publicly available vulnerability resources and references
Common Attack Pattern Enumeration and Classification (CAPEC™) - list of common attack patterns	Malware Attribute Enumeration and Characterization (MAEC™) - standardized language for attribute-based malware characterization	NIST Security Content Automation Protocol (SCAP) - security content for automating technical control compliance activities, vulnerability checking, and security measurement
Common Configuration Enumeration (CCE™) - common security configuration identifiers	Benchmark Development - resources for creating standards-based, structured, and automatable security guidance	Red Hat Repository - OVAL Patch Definitions corresponding to Red Hat Errata security advisories
Common Platform Enumeration (CPE™) - common platform identifiers	OVAL Interpreter - free tool for collecting information for testing, carrying out OVAL Definitions, and presenting results of the tests	Novell Repository - OVAL Definitions for SUSE Linux Enterprise compliance checking
CWE/SANS Top 25 - consensus list of the 25 most dangerous programming errors	Benchmark Editor™ - free tool that enhances and simplifies creation and editing of benchmark documents written in XCCDF and OVAL	Debian Repository - OVAL Definitions corresponding to Debian security advisories
Center for Internet Security (CIS) Consensus Security Metrics Definitions - set of standard metrics and data definitions that can be used across organizations to collect and analyze data on security process performance and outcomes	Recommendation Tracker™ - free tool that facilitates the development of automated security benchmarks	National Checklist Program Repository - U.S. government repository of publicly available security checklists/benchmarks
Twenty Most Important Controls and Metrics for Effective Cyber Defense and Continuous FISMA Compliance - twenty key actions or security "controls" that organizations must take to block or mitigate known and reasonably expected attacks	Extensible Configuration Checklist Description Format (XCCDF) - specification language for uniform expression of security checklists, benchmarks, and other configuration guidance	Center for Internet Security (CIS) Benchmarks - best-practice security configurations accepted for compliance with FISMA, the ISO standard, GLB, SOX, HIPAA, and FISMA, and other regulatory requirements for information security
SANS Top Twenty - SANS/FBI consensus list of the Twenty Most Critical Internet Security Vulnerabilities that uses CVE-IDs to identify the issues	Open Checklist Interactive Language (OCIL) - standardized language for expressing and evaluating non-automated security checks	DOD Security Technical Implementation Guides (STIGs) - U.S. Defense Information Systems Agency's (DISA) STIGs are configuration standards for DOD information assurance and information assurance-enabled devices and systems
OWASP Top Ten - ten most critical Web application security flaws	Common Vulnerability Scoring System (CVSS) - open standard that conveys vulnerability severity and helps determine urgency and priority of response	Common Frameworks for Vulnerability Disclosure and Response (CFVR) - standard format for reporting and sharing vulnerability information among multiple organizations
WASC Web Security Threat Classification - list of Web security threats	Policy Language for Assessment Results Reporting (PLAR) - language for requesting IT asset assessment results from tools, databases, and other products	Federal Desktop Core Configuration (FDOC) - CMS-mandated security configuration for Microsoft Windows Vista and XP operating system software
	Assessment Results Format (ARF) - open language for exchanging per-device assessment results data between assessment tools, asset databases, and other products that manage asset information	United States Government Configuration Baseline (USOCB) - security configuration baselines for IT products deployed across federal agencies
	Assessment Summary Results (ASR) - language for exchanging summarized assessment results data	

View the current collection of organizations, activities, and initiatives.

Disclaimer

This Web site is hosted by The MITRE Corporation. © 2010 The MITRE Corporation. CVE and OVAL are registered trademarks and the Making Security Measurable logo, CCE, CWE, CPE, CAPEC, CEE, MAEC, Benchmark Editor, and Recommendation Tracker are trademarks of The MITRE Corporation. All other trademarks are the property of their respective owners. Contact us: [measurablesecurity@mitre.org](mailto:measurablesecurity@mitre.org).

Page Last Updated: September 02, 2010

© 2010 The MITRE Corporation. All rights reserved.



# Questions?

[ramartin@mitre.org](mailto:ramartin@mitre.org)