Introduction to CVE, CWE, and the Top 25

Steve Christey Coley

October 29, 2015

@sushidude

coley@mitre.org

MITRE

Who Am I?

- **MITRE employee for 25+ years (so far)**
- **Started in artificial intelligence**
- **Like many others – fell into computer security**
- **Realized it's a great fit**
  - Always changing
  - Always challenging
  - Many opportunities to (try to) do the right thing
- **"MITRE partners with the government applying systems engineering and advanced technology to address issues of critical national importance."**
  - Values: Commitment to the Public Interest, People in Partnership, Excellence that Counts
  - Top STEM Company for Women, March 2015
  - Top Employer (Workforce Diversity for Engineering & IT Professionals Magazine)
  - Top STEM Employer (Hispanic Network Magazine)
- **http://www.mitre.org/about/mission-and-values**

Today's Theme

There is always a well-known solution to every human problem – neat, plausible, and wrong.

H.L. Mencken

Sometimes the simple life

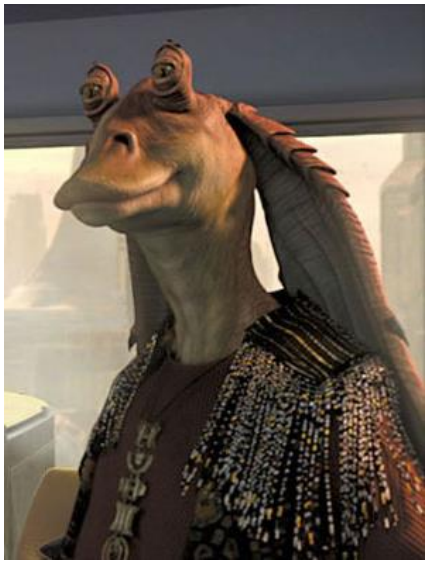
Ain't so simple.

1980's Van Halen

(the correct lineup)

Ye Goode Olde Dayes of 1999: Historical Context

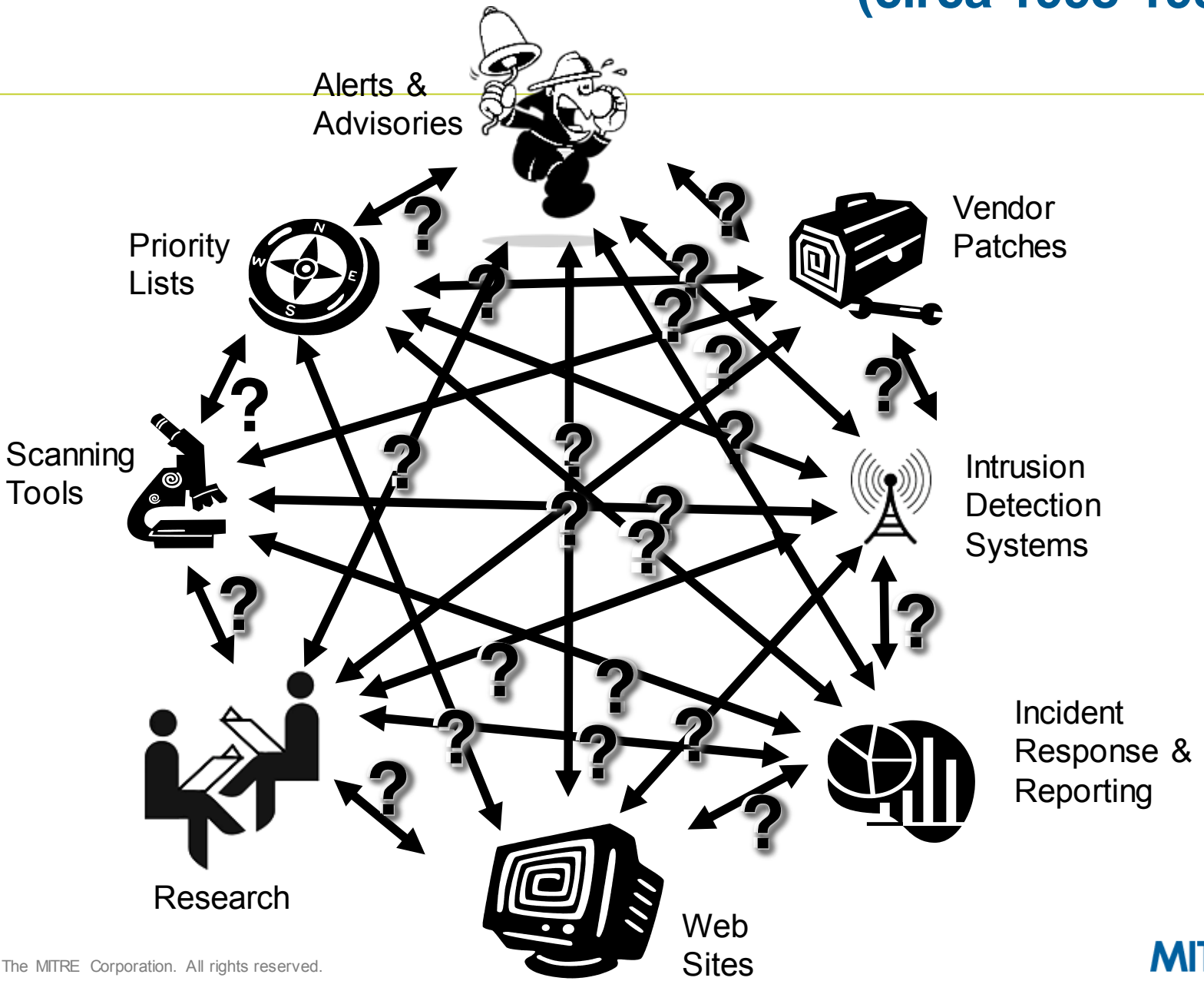
- Melissa worm
- The year before Y2K
- Bill Clinton impeached
- Euro currency established
- Wayne Gretzky retires
- SpongeBob Squarepants debuts
- Chandler Riggs (Carl from The Walking Dead) born
- Star Wars: The Phantom Menace introduces the world to Jar Jar Binks



Welcome to 1998

- **Vulnerability databases were mostly private**
  - "We'll show you our database if you show us your NDA"
- **Bugtraq was a low-traffic list**
- **Full-disclosure and OSVDB didn't exist**
- **CERT advisories said very little**
- **Exploits were shared privately**
- **Attacks were rampant for months/years**
- **Vendors didn't fix things for months/years**
- **Vulnerability scanning industry still in infancy**
- **WWW wasn't ubiquitous**
- **Maybe 10 unique vulnerability types**
- **"Smashing the Stack" was only 2 years old**
- **Most reported vulnerabilities were in servers**

Vulnerability Information Sharing (circa 1998-1999)



MITRE

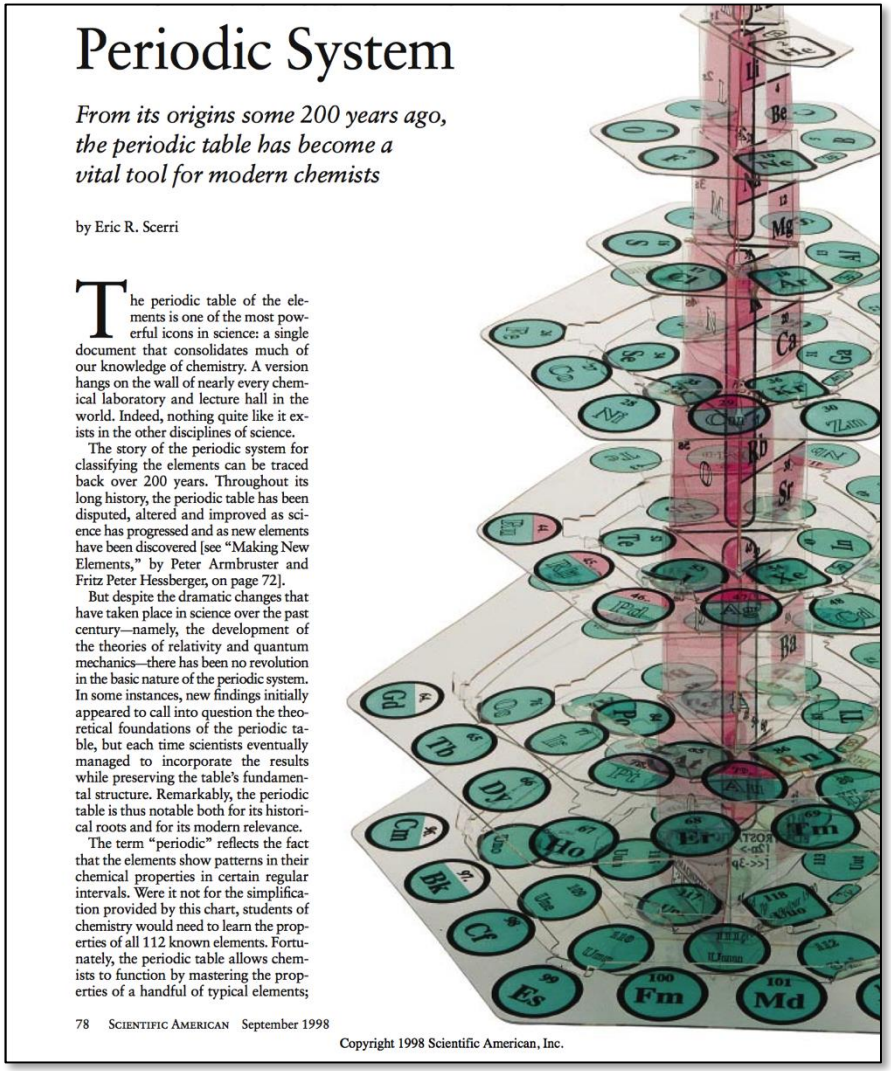
CVE Began with a Challenge at MITRE for 2 of our Technical Staff…

(Vulnerability Management: Circa 1998-1999)

- **How to pick a vulnerability scanning tool?**
  - Which one finds more?

- **Are we safe against vulnerabilities listed in CERT advisories?**
  - How to match CERT names of vulnerabilities with scanning tool results?

Aha moment on an Exercise Bike in MITRE's Bedford Fitness Center in 1998…



Periodic System

From its origins some 200 years ago, the periodic table has become a vital tool for modern chemists

by Eric R. Scerri

The periodic table of the elements is one of the most powerful icons in science: a single document that consolidates much of our knowledge of chemistry. A version hangs on the wall of nearly every chemical laboratory and lecture hall in the world. Indeed, nothing quite like it exists in the other disciplines of science.

The story of the periodic system for classifying the elements can be traced back over 200 years. Throughout its long history, the periodic table has been disputed, altered and improved as science has progressed and as new elements have been discovered [see "Making New Elements," by Peter Armbruster and Fritz Peter Hessberger, on page 72].

But despite the dramatic changes that have taken place in science over the past century—namely, the development of the theories of relativity and quantum mechanics—there has been no revolution in the basic nature of the periodic system. In some instances, new findings initially appeared to call into question the theoretical foundations of the periodic table, but each time scientists eventually managed to incorporate the results while preserving the table's fundamental structure. Remarkably, the periodic table is thus notable both for its historical roots and for its modern relevance.

The term "periodic" reflects the fact that the elements show patterns in their chemical properties in certain regular intervals. Were it not for the simplification provided by this chart, students of chemistry would need to learn the properties of all 112 known elements. Fortunately, the periodic table allows chemists to function by mastering the properties of a handful of typical elements;

78 Scientific American September 1998

Copyright 1998 Scientific American, Inc.

September 1998 Issue of Scientific American article on the Periodic System:

List of Elements predated the Periodic Table by 100's of Years

MITRE

2nd Workshop on Research with Security Vulnerability Databases, Purdue University

CERIAS
Center for Education and Research in Information Assurance and Security

Towards a Common Enumeration of Vulnerabilities

David E. Mann, Steven M. Christey
The MITRE Corporation
202 Burlington Rd., Bedford MA 01730
{damann, coley}@mitre.org

January 8, 1999

Abstract

In this paper, we discuss the use of multiple vulnerability databases in our oper
enterprise security environment and we consider some of the roadblocks we see
achieving interoperability between them. We introduce the concept of a Comm
Vulnerability Enumeration (CVE) as a mechanism that we believe will help to
easier data sharing. We consider some historical examples of the development
taxonomies in other fields and relate them to current efforts in representing and
vulnerability information. We present a simplified representation of a "vulneral
discuss how we anticipate using it to mitigate the problem of interoperability. V
describe some of the practical issues that may be involved in the development a
a CVE.

The Development of a Common ~~Vulnerability Enumeration~~ Vulnerabilities and Exposures List

Steven M. Christey
David W. Baker
William H. Hill
David E. Mann

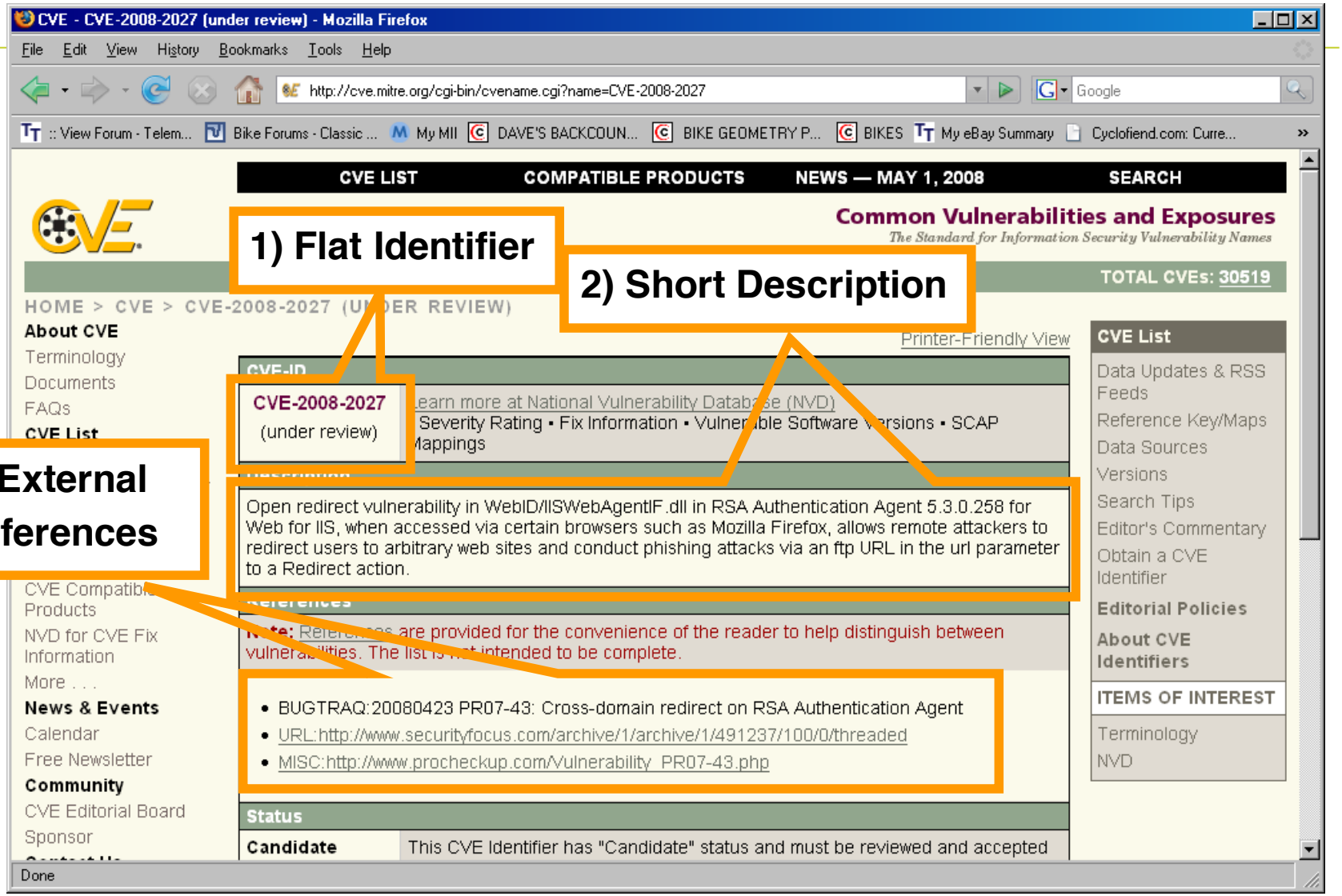
The MITRE Corporation

MITRE

CVE Editorial Board

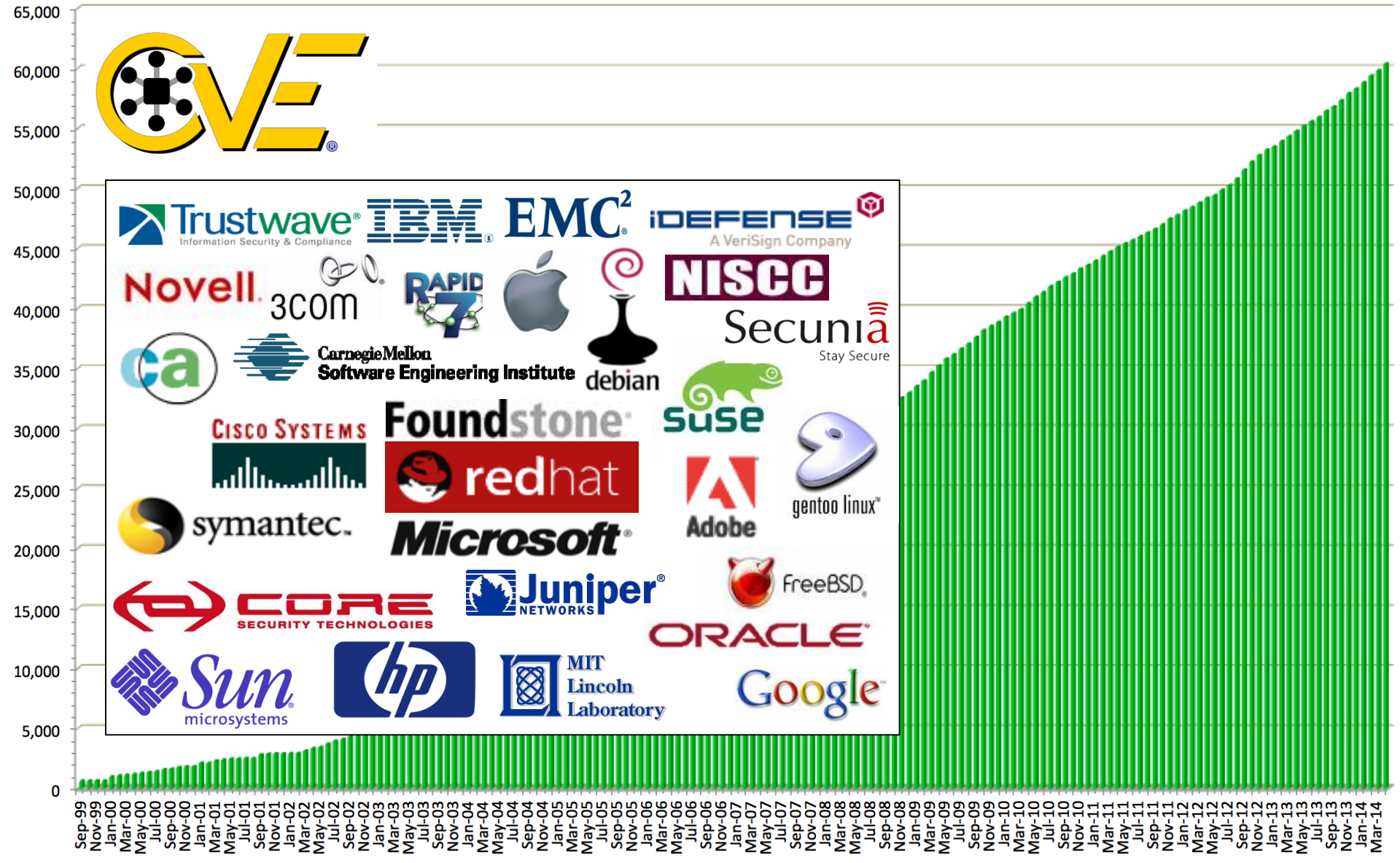
MITRE, Sun microsystems, NSA, CITADEL SECURITY SOFTWARE, IBM, NIST National Institute of Standards and Technology, BIND VIEW, GNSC, NFR SECURITY, STAT, Microsoft, redhat, U.S. Department of Energy CIAC Computer Incident Advisory Center, Carnegie Mellon Software Engineering Institute, CERIAS, SANS INSTITUTE, INTERNET SECURITY SYSTEMS, symantec, Cisco Systems, INTRANODE, SILICON DEFENSE, EWA, nCircle NETWORK SECURITY, ca Computer Associates, security-focus.com, Nessus, UCDAVIS, GENERAL DYNAMICS Advanced Information Systems, GUARDEDNET

CVE Entries: Dictionary, not a Database



MITRE

CVE 1999 to 2014



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**

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

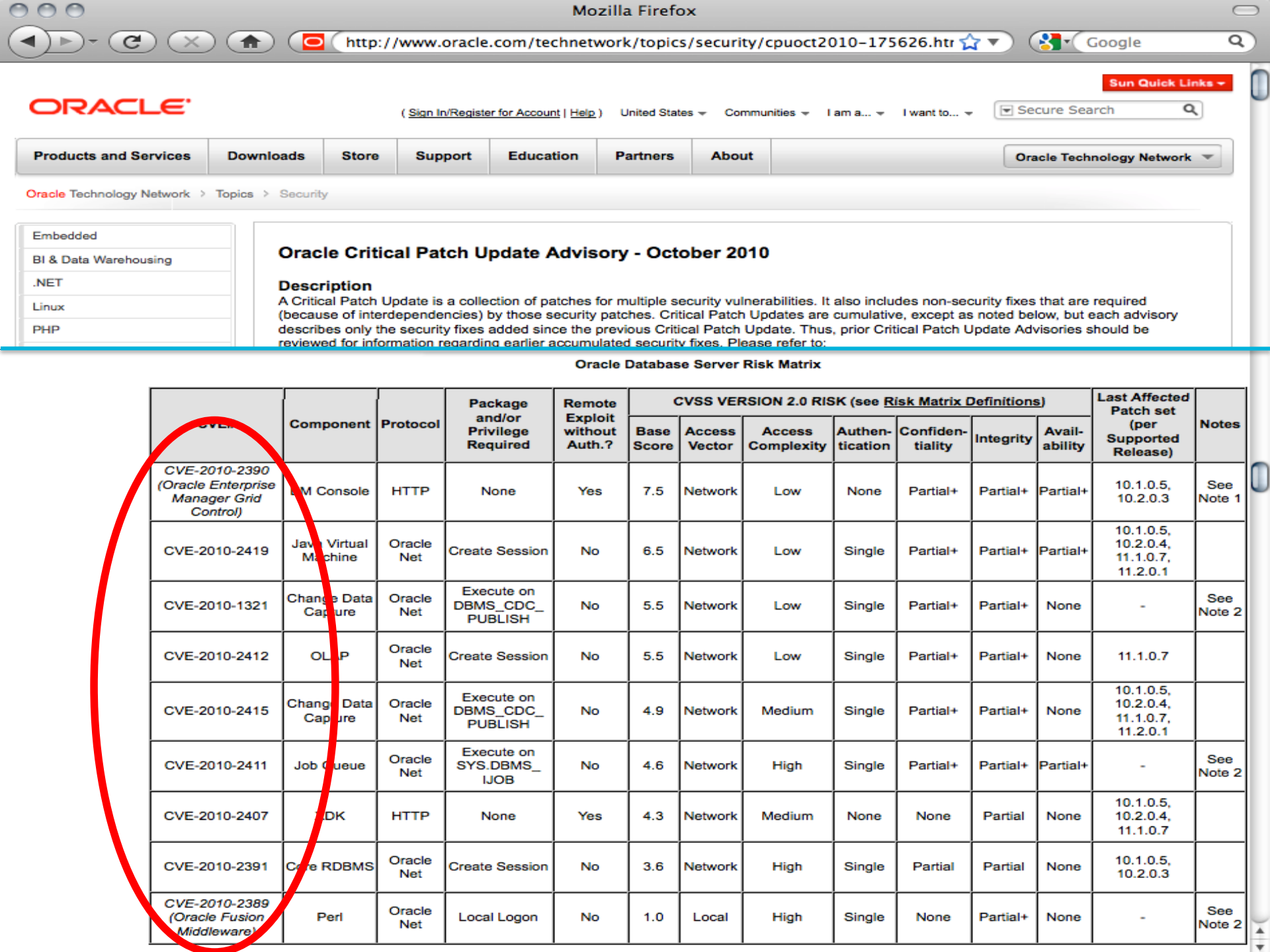
Subject: **APPLE-SA-2010-08-11-1 iOS 4.0.2 Update for iPhone and iPod touch**
From: Apple Product Security <email@hidden>
Date: Wed, 11 Aug 2010 12:19:43 -0700
Delivered-to: email@hidden
Delivered-to: email@hidden

```
-----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
```

redhat

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



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	JDK	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

Mission and Overview

NVD is the U.S. government repository of standards based vulnerability management data. This data enables automation of vulnerability management, security measurement, and compliance (e.g. FISMA).

Resource Status

NVD contains:

62403 CVE Vulnerabilities
231 Checklists
248 US-CERT Alerts
2867 US-CERT Vuln Notes
10286 OVAL Queries
90649 CPE Names

Last updated: 5/23/2014 5:36:54 PM

CVE Publication rate: 19.4

Email List

NVD provides four mailing lists to the public. For information and subscription instructions please visit NVD Mailing Lists

Workload Index

Vulnerability Workload Index: 8.19

About Us

NVD is a product of the

National Cyber Awareness System

Vulnerability Summary for CVE-2014-0160

Original release date: 04/07/2014

Last revised: 05/23/2014

Source: US-CERT/NIST

Overview

The (1) TLS and (2) DTLS implementati
information from process memory via cr
bug.

Impact

CVSS Severity (version 2.0):

CVSS v2 Base Score: 5.0 (MEDIUM) (

Impact Subscore: 2.9

Exploitability Subscore: 10.0

CVSS Version 2 Metrics:

Access Vector: Network exploitable

Access Complexity: Low

Authentication: Not required to explo

Impact Type: Allows unauthorized dis

CVSS V2 scoring evaluates the impact c
into account the nature of the data that
memory on the targeted host, a succes
cryptographic keys and passwords. The
and functions of that system.

References to Advisories,

By selecting these links, you will be leav
inferences should be drawn on account
not necessarily endorse the views expre
these sites. Please address comments a

References to Advisories, Solutions, and Tools

By selecting these links, you will be leaving NIST webspace. We have provided these links to other web sites because they may have information that would be of interest to you. No inferences should be drawn on account of other sites being referenced, or not, from this page. There may be other web sites that are more appropriate for your purpose. NIST does not necessarily endorse the views expressed, or concur with the facts presented on these sites. Further, NIST does not endorse any commercial products that may be mentioned on these sites. Please address comments about this page to nvd@nist.gov.

External Source: SECTRACK
Name: 1030077
Hyperlink: http://www.securitytracker.com/id/1030077

External Source: FULLDISC
Name: 20140411 MRI Rubies may contain statically linked, vulnerable OpenSSL
Hyperlink: http://seclists.org/fulldisclosure/2014/Apr/173

External Source: HP
Name: HPSBMU02995
Hyperlink: http://marc.info/?l=bugtraq&m=139722163017074&w=2

External Source: MISC
Name: https://www.cert.fi/en/reports/2014/vulnerability788210.html
Hyperlink: https://www.cert.fi/en/reports/2014/vulnerability788210.html

External Source: CONFIRM
Name: http://www.oracle.com/technetwork/topics/security/opensslheartbleedcve-2014-0160-2188454.html
Hyperlink: http://www.oracle.com/technetwork/topics/security/opensslheartbleedcve-2014-0160-2188454.html

External Source: CONFIRM
Name: http://www-01.ibm.com/support/docview.wss?uid=swg21670161
Hyperlink: http://www-01.ibm.com/support/docview.wss?uid=swg21670161

External Source: REDHAT
Name: RHSA-2014:0378
Hyperlink: http://rhn.redhat.com/errata/RHSA-2014-0378.html

External Source: CONFIRM
Name: http://www.f-secure.com/en/web/labs_global/fsc-2014-1
Hyperlink: http://www.f-secure.com/en/web/labs_global/fsc-2014-1

External Source: CONFIRM
Name: http://www.splunk.com/view/SP-CAAAMB3
Hyperlink: http://www.splunk.com/view/SP-CAAAMB3

Vulnerability Information Sharing (circa 1999+)

Content Decisions

- **Explicit guidelines for content of CVE entries**
  - Ensure and publicize consistency within CVE
  - Provide “lessons learned” for researchers
  - Document differences between vulnerability “views”
- **Two basic types**
  - Inclusion: What goes into CVE? What doesn’t, and why?
  - Level of Abstraction: One or many entries for similar issues?
  - Format: How are CVE entries formatted?
- **Difficult to document**
  - “[It’s] like trying to grasp wet corn starch” (Board member)

Incomplete information is the bane of consistency - and content decisions!

Why CVE-2001-0019 Could Identify 1, 2, or 6 Vulnerabilities

- "Shellshock" anyone?
- 3 different source code scenarios
- Without actual source, can't be sure which scenario is true
- Even with source, there are different ways of counting
- Multiple format string problems are especially difficult to distinguish

```
strcpy(arg, long_input);
if (strcmp(cmd, "show") == 0) {
  process_show_command(arg); }
elsif (strcmp(cmd, "clear") == 0) {
  process_show_command(arg); }
```

```
if (strcmp(cmd, "show") == 0) {
 strcpy(str, long_input);
 process_show_command(str); }
elsif (strcmp(cmd, "clear") == 0) {
 strcpy(str, long_input);
 process_clear_command(str); }
```

```
if (strcmp(cmd, "show") == 0) {
  if (strcmp(arg1, "script") == 0) {
     strcpy(str, long_input);
     show_script(str); }
  elsif (strcmp(arg1, "archive") == 0) {
     strcpy(str, long_input);
     show_archive(str); }
  elsif (strcmp(arg1, "log") == 0) {
     strcpy(str, long_input);
     show_log(str); } }
elsif (strcmp(cmd, "clear") == 0) {
  if (strcmp(arg1, "script") == 0) {
    strcpy(str, long_input);
    show_script(str); }
  elsif (strcmp(arg1, "archive") == 0) {
    strcpy(str, long_input);
    show_archive(str); }
  elsif (strcmp(arg1, "log") == 0) {
    strcpy(str, long_input);
    show_log(str); } }
```

VDB Abstraction: 1 to 5 Entries?

CVE-1: SQL injection in version 1.x through login.php and order.php.

CVE-2: SQL injection in version 2.x through admin.php.

CVE-3: XSS in version 2.x through login.php and search.php.

ISS and Bugtraq ID

1: Mult. SQL injection in 1.x and 2.x

2: XSS in 2.x

Secunia, ISS, and Bugtraq ID

1: SQL injection and XSS in 1.x and 2.x

Somebody somewhere, probably

1: login.php

2: order.php

3: admin.php

4: search.php

OSVDB

1: SQL injection in login.php

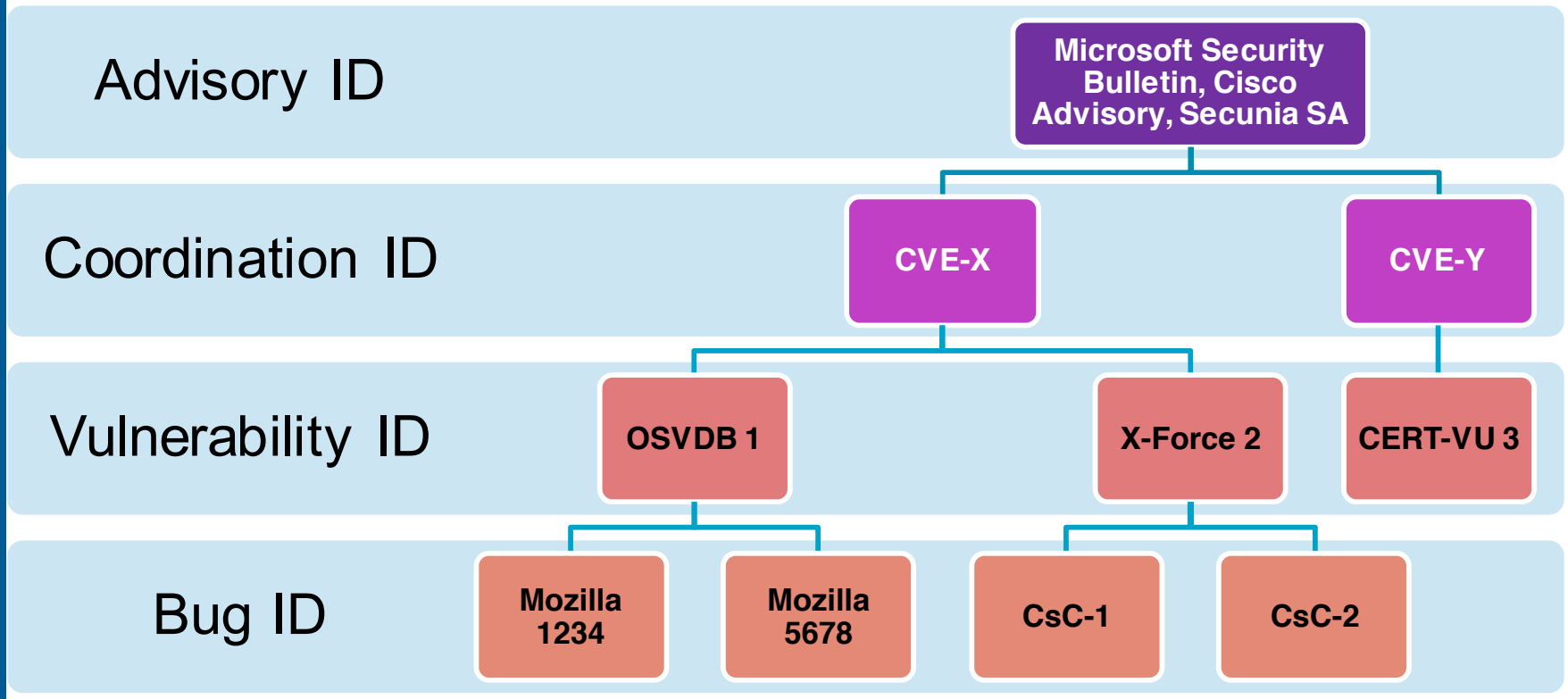
2: SQL injection in order.php

3: SQL injection in admin.php

4: XSS in login.php

5: XSS in search.php

Different Audience → Different Abstraction



- *CVE was always intended as a coordination ID*
- *We originally thought that coordination could operate at the vulnerability level*
- *But, there's too much fluctuation and variation in vulnerability information in the early stages, when coordination ID is most needed*

MITRE

Content Decisions: Abstraction

- AB1: SPLIT if different flaw types
- AB2: SPLIT if different versions are affected
- SPLIT if different vectors are released at a later time
- SPLIT if different codebases
- Otherwise MERGE
- Refinements and/or interpretations of the above

These factors are generally stable across all phases of vulnerability disclosure, and often known early in the game.

http://cve.mitre.org/cve/editorial_policies/cd_abstraction.html

Content Decisions: Inclusion

- **INCLUDE any issue for software that**
  - Could be deployed in an enterprise
  - Could be network-connected physical devices
  - Has minimal, but non-zero, risk
    - path disclosure, admin-to-SYSTEM, client-side crasher
- **EXCLUDE any issue that**
  - Is “site-specific,” SaaS, hosted, “in the cloud,” …
  - Is provably wrong
  - Is just a rumor
  - Is not “actionable”
  - Is “just a bug” (e.g. defenestration exploit)

Site-specific / hosted software can be difficult to identify.

Issue: What is a Vulnerability?

- **CVE was originally called "Common Vulnerability Enumeration"**
- **Security tools included many "non-vulnerabilities"**
- **"Terminological warfare" by Editorial Board in August 1999**
  - 2 main debates
    - What is a vulnerability?
    - Should CVE include things that aren't vulnerabilities?
  - Primary example: running finger (CVE-1999-0612)
    - "Stepping stone" but not directly exploitable
  - Various alternate terms were debated
  - "Exposure" wasn't being used that often back then, and there was a strong need to keep the CVE acronym, so...
- **See:**
  - http://cve.mitre.org/about/terminology.html
  - http://cve.mitre.org/board/archives/1999-08/threads.html

Vulnerability definitions vary widely!

Issue: What is a Real Vulnerability?

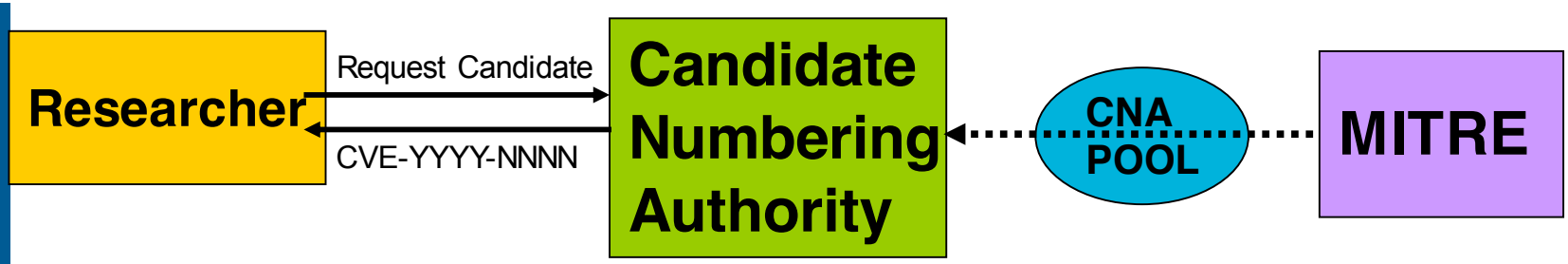
- **~50% of all issues are not publicly acknowledged by the vendor**
  - http://cve.mitre.org/board/archives/2000-09/msg00038.html
- **Many vulnerabilities are found in obscure software by unknown researchers without independent confirmation**
- **Resource-intensive to verify every report**
- **Some issues don't cross "privilege boundaries"**
- **Some issues are technically security issues, but extremely low risk**
- **If it's reported but it may not be real, should it be added to CVE?**
  - It will at least be reviewed
  - How much verification is necessary?

0 **Extreme example**

CVE-1999-0205	Denial of service in Sendmail 8.6.11 and 8.6.12

- **Could not be replicated by vendor**
- **Checked by multiple tools (which may only compare banners)**

Candidate Reservation Process



- Request candidate from CNA
- Provide candidate number to vendor and other parties
- Include candidate number in initial public announcement
- Notify MITRE of announcement
- Perform due diligence to avoid duplicate or incorrect candidates
- Should work with affected vendor to increase confidence in correctness of the candidate

- Obtain pool of candidate numbers from MITRE
- Define requirements for researchers to obtain a candidate
- Assign correct number of candidate numbers
- Ensure candidate is shared across all parties
- Do not use candidates in "competitive" fashion

- Primary CNA
- Accessible to researchers via cve-assign@mitre.org
- Educate CNA about content decisions
- Update CVE web site when candidate is publicly announced
- Track potential abuses

Anatomy of a CVE Description: CVE-2009-4623

Multiple PHP remote file inclusion vulnerabilities in Advanced Comment System 1.0 allow remote attackers to execute arbitrary PHP code via a URL in the ACS_path parameter to (1) index.php and (2) admin.php in advanced_comment_system/. NOTE: this might only be a vulnerability when the administrator has not followed installation instructions in install.php.

Flaw type, vendor name, product name, affected versions, remote/local, impact, attack vectors, clarifiers.

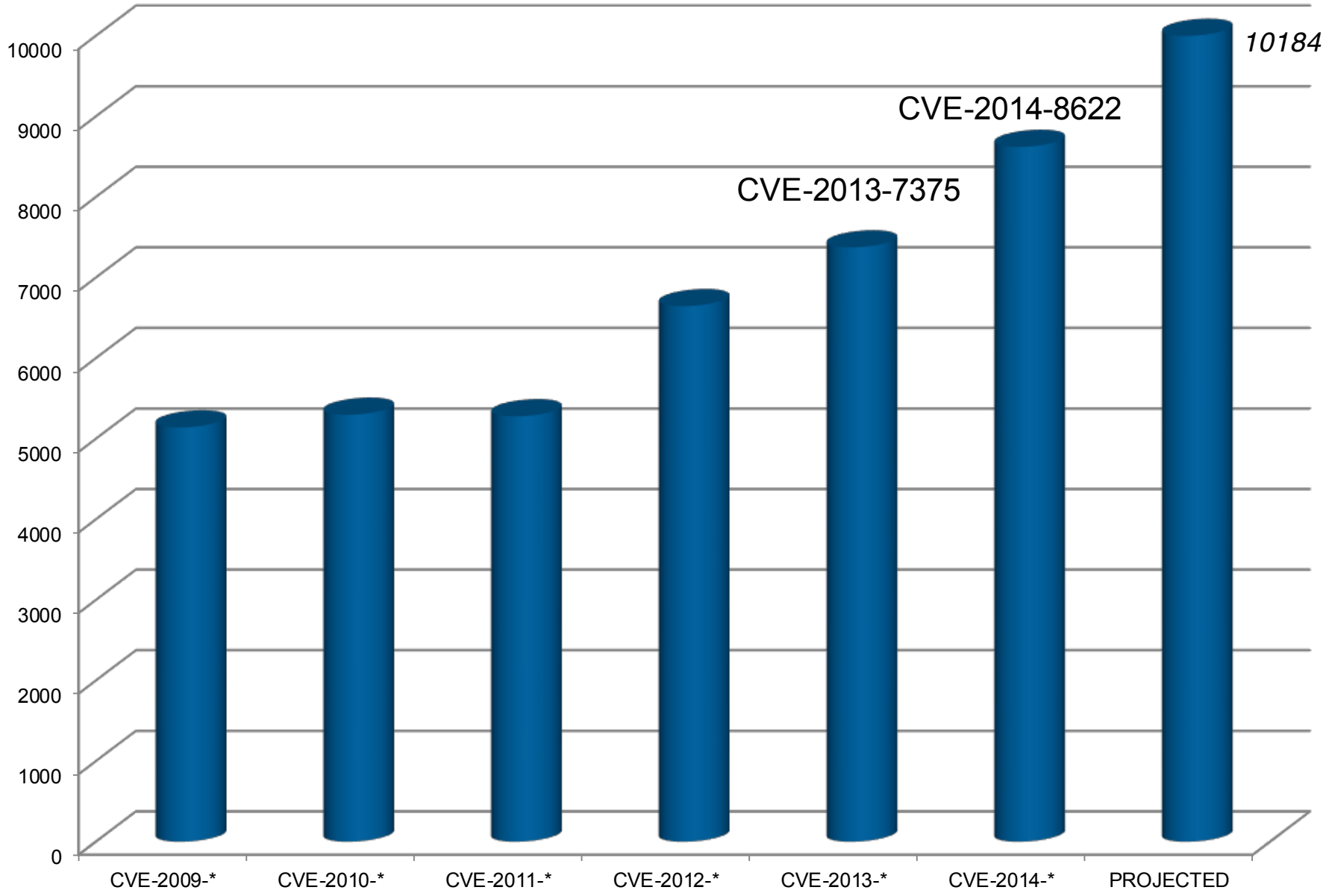
10 Years of CVE Descriptions

CVE	Desc
CVE-1999-0067	CGI phf program allows remote command execution through shell metacharacters.
CVE-2000-0067	CyberCash Merchant Connection Kit (MCK) allows local users to modify files via a symlink attack.
CVE-2001-0067	The installation of J-Pilot creates the .jpilot directory with the user's umask, which could allow local attackers to read other users' PalmOS backup information if their umasks are not securely set.
CVE-2002-0067	Squid 2.4 STABLE3 and earlier does not properly disable HTCP, even when "htcp_port 0" is specified in squid.conf, which could allow remote attackers to bypass intended access restrictions.
CVE-2003-0067	The aterm terminal emulator 0.42 allows attackers to modify the window title via a certain character escape sequence and then insert it back to the command line in the user's terminal, e.g. when the user views a file containing the malicious sequence, which could allow the attacker to execute arbitrary commands.
CVE-2004-0067	Multiple cross-site scripting (XSS) vulnerabilities in phpGedView before 2.65 allow remote attackers to inject arbitrary HTML or web script via (1) descendancy.php, (2) index.php, (3) individual.php, (4) login.php, (5) relationship.php, (6) source.php, (7) imageview.php, (8) calendar.php, (9) gedrecord.php, (10) login.php, and (11) gdbi_interface.php. NOTE: some aspects of vector 10 were later reported to affect 4.1.

10 Years of CVE Descriptions

CVE	Desc
CVE-2005-0067	The original design of TCP does not require that port numbers be assigned randomly (aka "Port randomization"), which makes it easier for attackers to forge ICMP error messages for specific TCP connections and cause a denial of service, as demonstrated using (1) blind connection-reset attacks with forged "Destination Unreachable" messages, (2) blind throughput-reduction attacks with forged "Source Quench" messages, or (3) blind throughput-reduction attacks with forged ICMP messages that cause the Path MTU to be reduced. NOTE: CVE-2004-0790, CVE-2004-0791, and CVE-2004-1060 have been SPLIT based on different attacks; CVE-2005-0065, CVE-2005-0066, CVE-2005-0067, and CVE-2005-0068 are related identifiers that are SPLIT based on the underlying vulnerability. While CVE normally SPLITs based on vulnerability, the attack-based identifiers exist due to the variety and number of affected implementations and solutions that address the attacks instead of the underlying vulnerabilities.
CVE-2006-0067	SQL injection vulnerability in login.php in VEGO Links Builder 2.00 and earlier allows remote attackers to execute arbitrary SQL commands via the username parameter.
CVE-2007-0067	Unspecified vulnerability in the Lotus Domino Web Server 6.0, 6.5.x before 6.5.6, and 7.0.x before 7.0.3 allows remote attackers to cause a denial of service (daemon crash) via requests for URLs that reference certain files.
CVE-2008-0067	Multiple stack-based buffer overflows in HP OpenView Network Node Manager (OV NNM) 7.01, 7.51, and 7.53 allow remote attackers to execute arbitrary code via (1) long string parameters to the OpenView5.exe CGI program; (2) a long string parameter to the OpenView5.exe CGI program, related to ov.dll; or a long string parameter to the (3) getcvdata.exe, (4) ovlaunch.exe, or (5) Toolbar.exe CGI program.
CVE-2009-0067	** RESERVED **
CVE-2010-0067	Unspecified vulnerability in the Oracle Containers for J2EE component in Oracle Application Server 10.1.2.3 and 10.1.3.4 allows remote attackers to affect confidentiality via unknown vectors.

Maximum CVE-YYYY-nnnn ID per year (as of Nov 5, 2014)



MITRE

We Have a CVE-10K Problem: What Do We Do After CVE-2014-9999?

The Register®

Data Center Software Networks **Security** Policy Business Hardware Science Bootnotes Columnis

SECURITY

Bug-hunters: They're coming outta the goddamn walls, aargh!

Security bods prep for more and more aliens bursting out of software

By John Leyden, 5 Feb 2013 Follow 2,596 followers

12

The organisation that administers the industry standard for classifying computer system security vulnerabilities wants to prepare its classification system for a world with an even greater number of bugs.

Mitre Corp is considering adding a 100 times more CVE (Common Vulnerabilities and Exposures) slots each year to accommodate bug reports.

The current syntax CVE-YYYY-NNNN supports up to 9,999 vulnerabilities. However the increasing number of software flaw

RELATED STORIES

Bug kills Intel gig-E controllers

Openistas squish security bugs twice as

MITRE

Yawn. So What?

- If we made a 4-digit assumption, maybe^Wdefinitely others did too
- A lot of code, processes, & formats use CVE IDs
- Hundreds of CVE-compatible products in many languages
- Thousands of “users” across the globe
- We don’t know where that all is

- **CVE is part of the infrastructure**
- **CVE is everywhere**
- **People depend on it without even knowing**
- **People use it in ways we don’t know**

- Obligatory Heartbleed (I mean, CVE-2014-0160) reference
  - Which obscure nooks and crannies of the Interwebz has it been found lately?

Where the Wild Things Are

- **Output Format**
  - Wider than 13-character columns
  - Sorting
- **Input Format**
  - Data lengths
  - Structures
  - Search routines
- **Extraction or Parsing**
  - 4-digit assumption, if violated, could trigger silent failure, fatal error, or use of the wrong ID for an unrelated vulnerability

Interpreters Don't Care ('bout number representation)

```
# My awesome CVE ID detector in Perl. Shush.
$str = "CVE-2014-839572957648549";
if ($str =~ /CVE-(\d+)-(\d+)/) {
   $id = sprintf("CVE-%4d-%04d", $1, $2);
}
else { $id = "PARSE-ERROR"; }
print "ID = $id\n";
```

CVE-2014--001

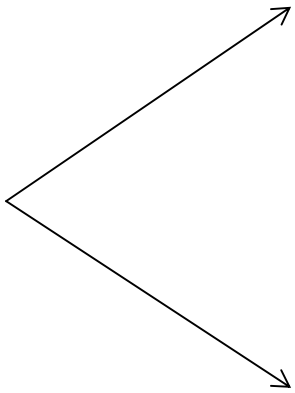
- Big number that sprintf can't handle? Return -1
- Format -1 with leading zeroes in 4 digits: -001

MITRE

Sorting

- CVE IDs aren't published in order, but good sorting is aesthetic and sometimes a good visual optimization
- What happens with typical string-only sorting of variable-length IDs?

CVE-2014-9999
CVE-2014-10000
CVE-2014-1234
CVE-2014-12345



CVE-2014-1234
CVE-2014-9999
CVE-2014-10000
CVE-2014-12345

CVE-2014-10000
CVE-2014-1234
CVE-2014-12345
CVE-2014-9999

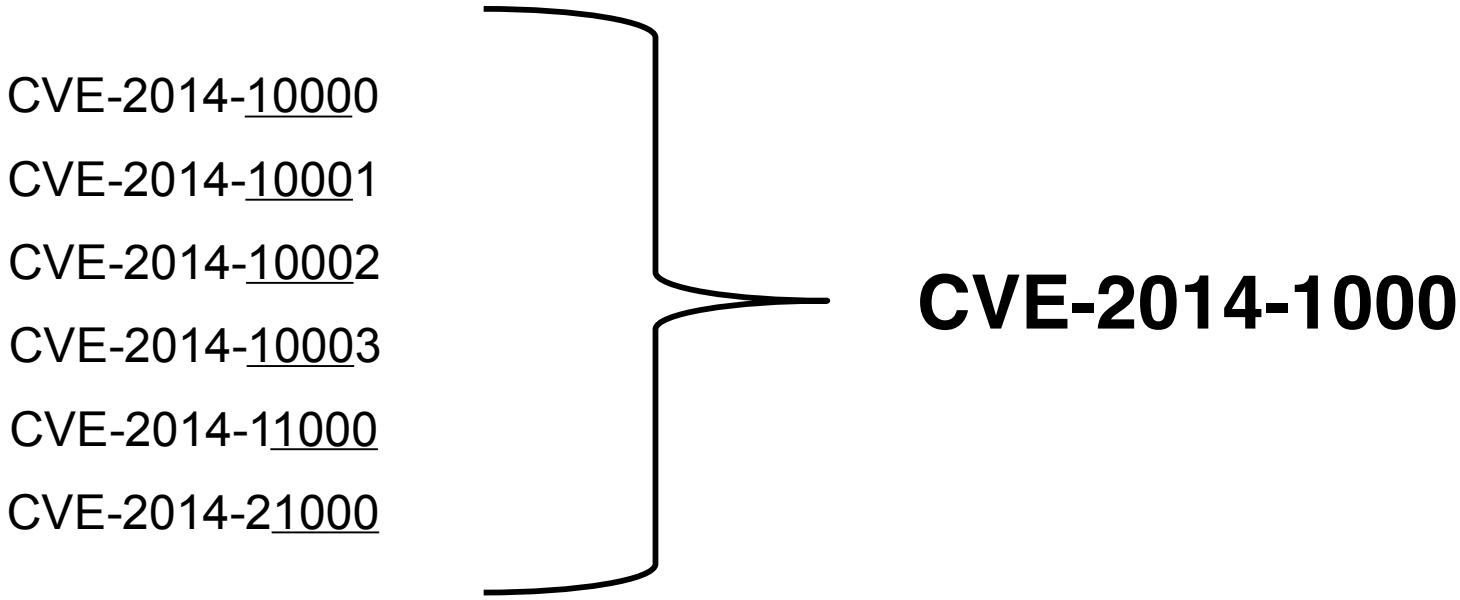
The New Syntax – Starting January 1, 2014

CVE-YYYY-NNNN…N

- 4-digit minimum in sequence number
- No maximum
- Add extra digits only when needed
- Only leading 0's with 4 digits

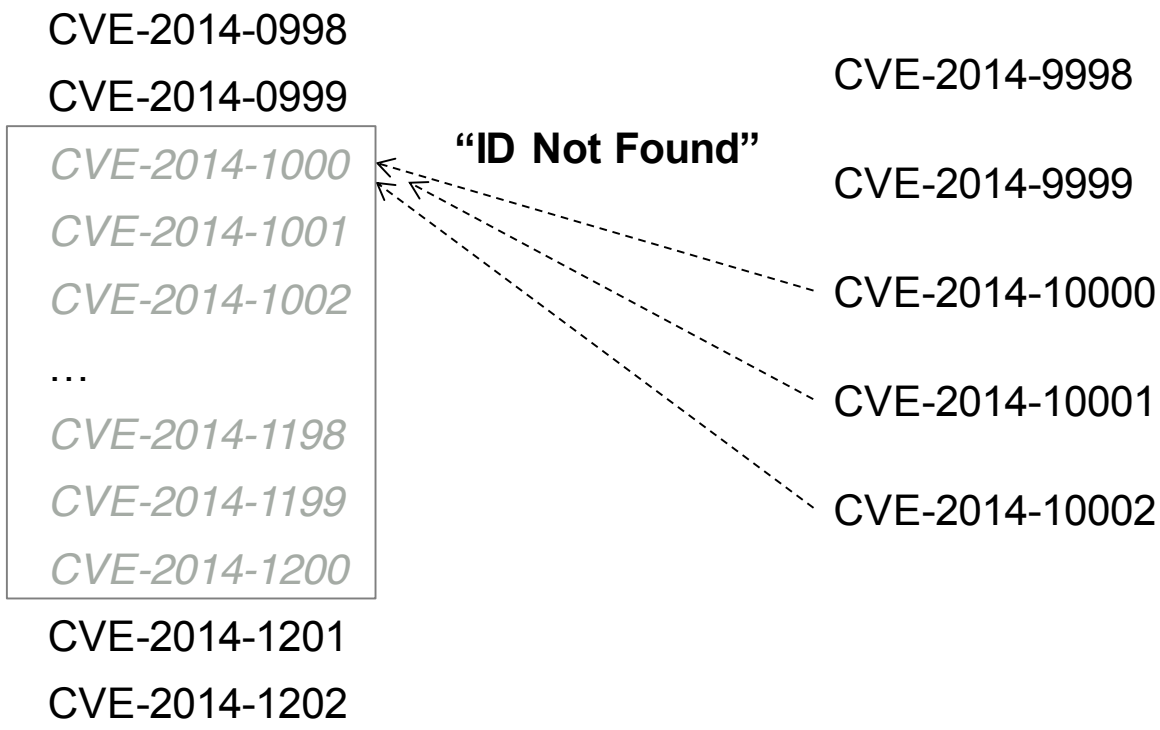
http://cve.mitre.org/cve/identifiers/syntaxchange.html

Truncation: The Four Digit Assumption



- Wrong ID = the wrong vulnerability = wasted time and, worse, being vulnerable and not knowing it!
- We have seen (and I have written) code that does truncation.
- We have seen at least one live web site that truncates

Minimizing the Pain of Truncation Errors: The Protection Block

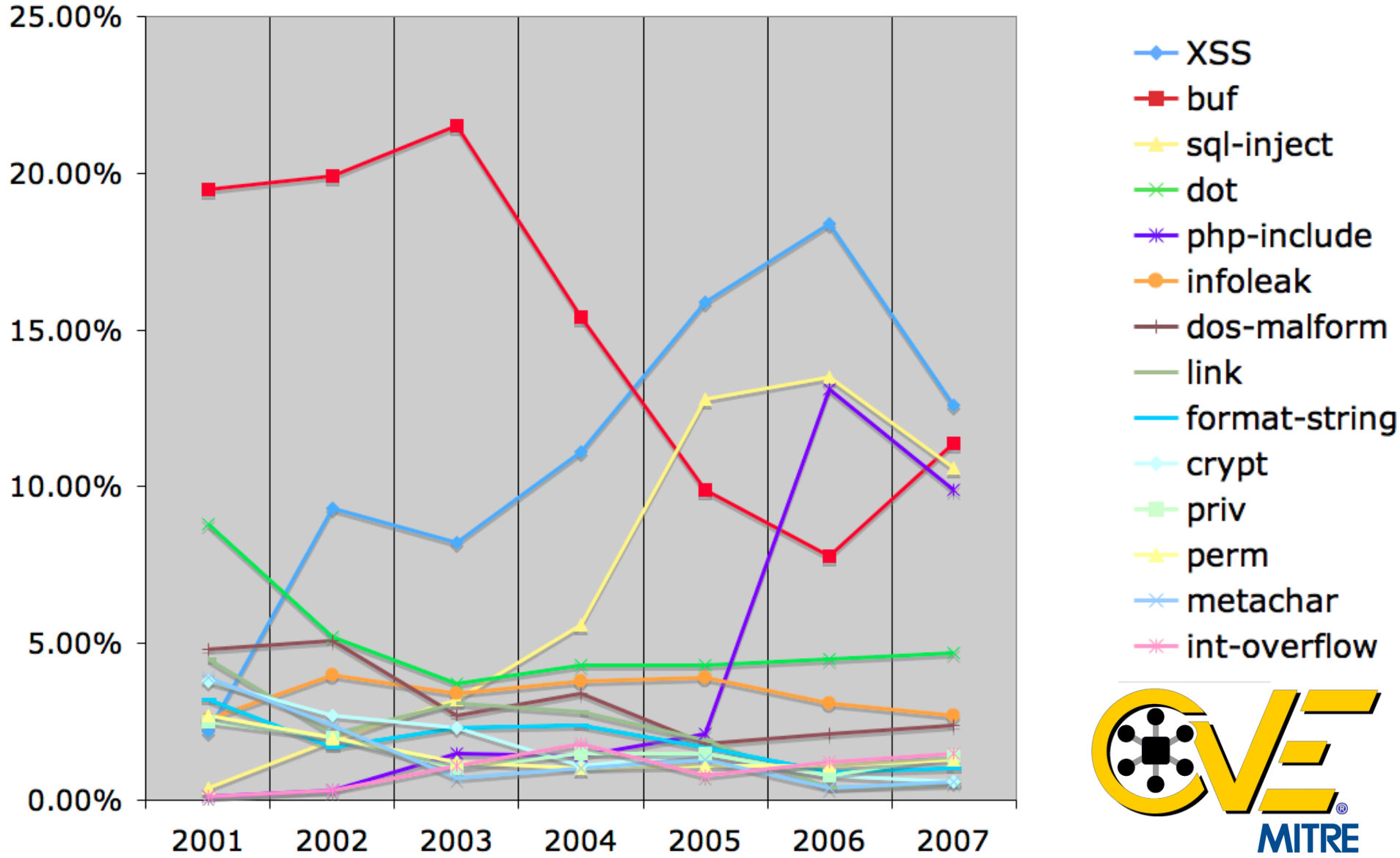


MITRE

From individual vulnerabilities to whole classes of problems…

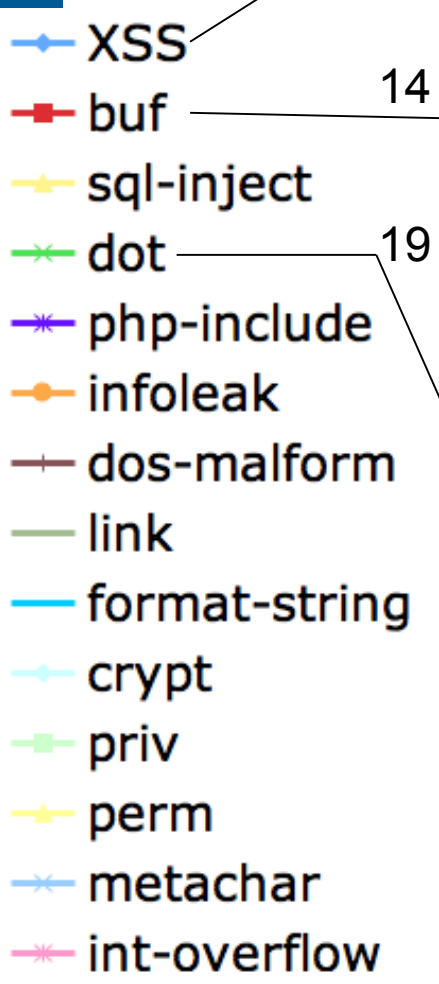
MITRE

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

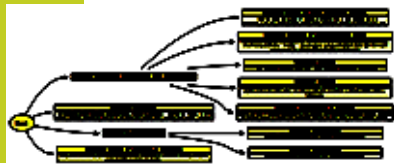
Improper Restriction of Operations within the Bounds of a Memory Buffer (119)
- Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') (120)
- Write-what-where Condition (123)
- Out-of-bounds Read (125)
- Improper Handling of Length Parameter Inconsistency (130)
- Improper Validation 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

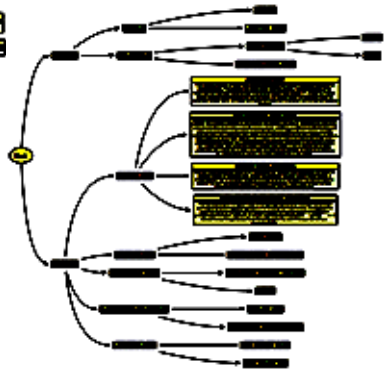
Path Traversal (22)
- Relative Path Traversal (23)
  - Path Traversal: '../filedir' (24)
  - Path Traversal: '/../filedir' (25)
  - <----------8 more here ----------->
  - Path Traversal: '....//' (34)
  - Path Traversal: '.../...//' (35)
- Absolute Path Traversal (36)
  - Path Traversal: '/absolute/pathname/here'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

CWE™ MITRE

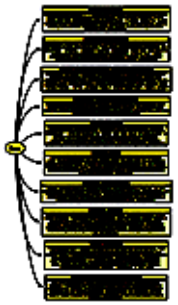
Protection Analysis



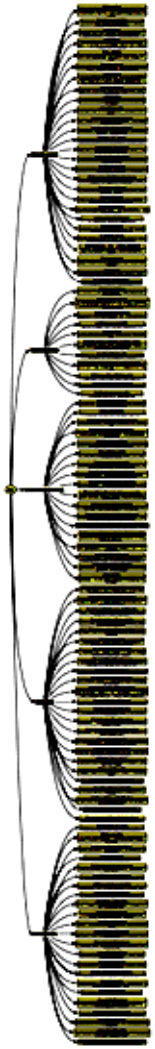
Weber



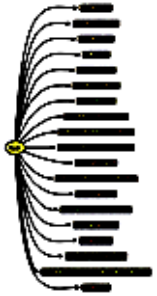
OWASP



CLASP

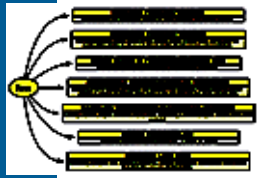


Microsoft

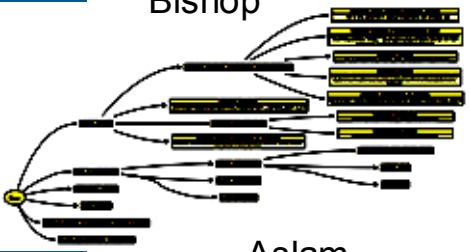


PLOVER

RISOS

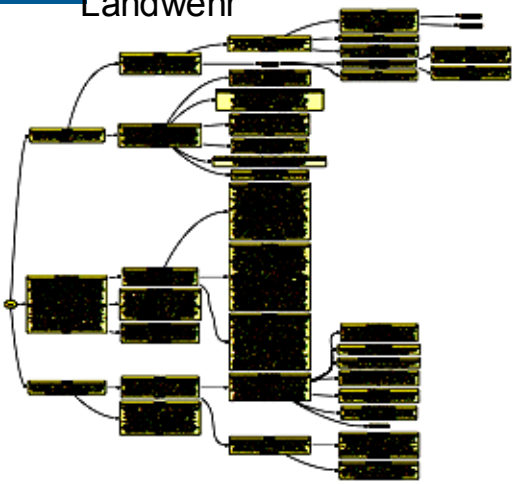


Bishop

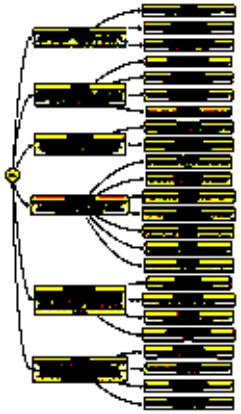


Aslam

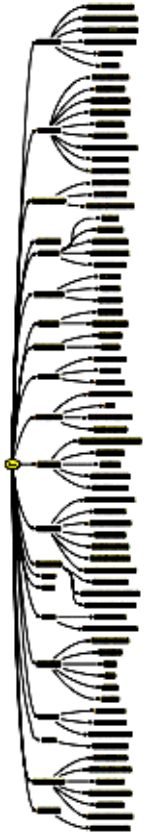
Landwehr



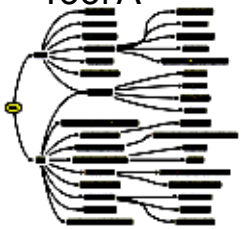
WASC



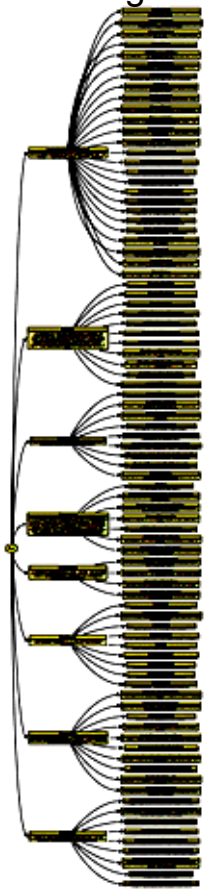
Tool B



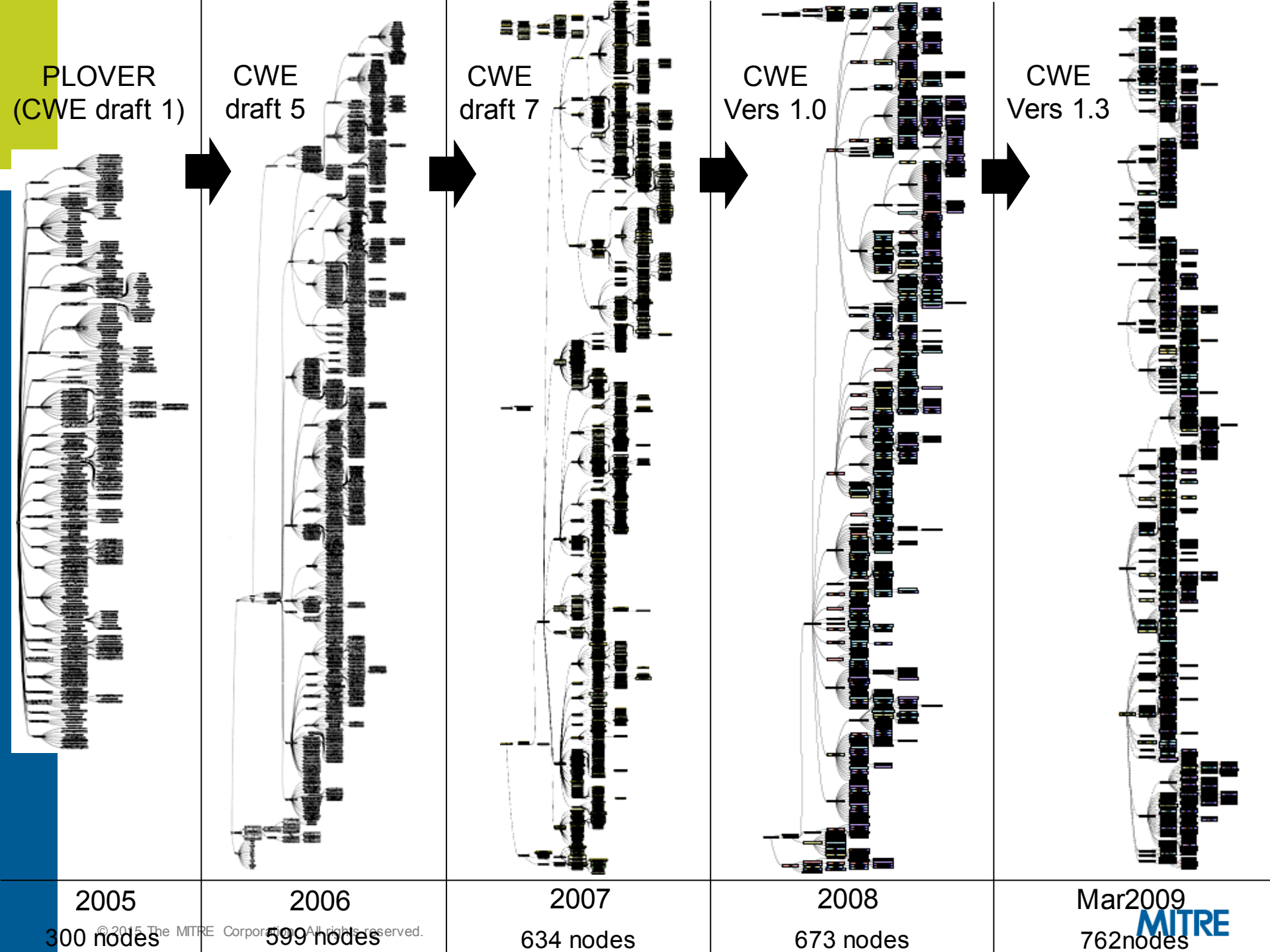
Tool A



7 Kingdoms



MITRE



Current Community Contributing to the Common Weakness Enumeration

- AppSIC
- Apple
- Aspect Security
- Booz Allen Hamilton Inc.
- Cenzic
- CERIAS/Purdue University
- CERT/CC
- Cigital
- Codenomicon
- Core Security
- Coverity
- DHS
- Fortify
- Gramma Tech
- IPA/JPCERT
- IBM
- Interoperability Clearing House
- JHU/APL
- JMU
- Kestrel Technology
- KDM Analytics
- Klocwork
- McAfee
- Microsoft
- MIT Lincoln Labs
- MITRE
- North Carolina State University
- NIST
- NSA
- OMG
- Oracle
- Ounce Labs
- OSD
- OWASP
- Palamida
- Parasoft
- PolySpace Technologies
- proServices Corporation
- SANS Institute
- SecurityInnovation
- Security University
- Semantic Designs
- SofCheck
- SPI Dynamics
- SureLogic, Inc.
- Symantec
- UNISYS
- VERACODE
- Watchfire
- WASC
- Whitehat Security, Inc.

To join send e-mail to cwe@mitre.org

Recent Posts

MS08-078 and the SDL
Announcing CAT.NET CTP and AntiXSS v3 beta
SDL videos
BlueHat SDL Sessions Wrap-up
Secure Coding Secrets?

Tags

Common Criteria **Crawl Walk Run**
Privacy **SDL** SDL Pro Network
Security Assurance Security Blackhat
SDL **threat modeling**

News

Blogroll

BlueHat Security Briefings
The Microsoft Security Response Center
Michael Howard's Web Log
The Data Privacy Imperative
Security Vulnerability Research & Defense
Visual Studio Code Analysis Blog
MSRC Ecosystem Strategy Team

Books / Papers / Guidance

The Security Development Lifecycle (Howard and Lipner)
Privacy Guidelines for Developing Software Products and Services
Microsoft Security Development Lifecycle (SDL) — Portal
Microsoft Security Development Lifecycle (SDL) — Process Guidance (Web)
Microsoft Security Development Lifecycle (SDL) — Process Guidance (.doc)

MS08-078 and the SDL ★★★★★

Hi, Michael here.

Every bug is an opportunity to learn, and the security update that fixed the data binding bug that affected Internet Explorer users is no exception.

The Common Vulnerabilities and Exposures (CVE) entry for this bug is CVE-2008-4844.

Before I get started, I want to explain the goals of the SDL and the security work here at Microsoft. The SDL is designed as a multi-layered process to help systemically reduce security vulnerabilities; if one component of the SDL process fails to prevent or catch a bug, then some other component should prevent or catch the bug. The SDL also mandates the use of security defenses whose impact will be reflected in the "mitigations" section of a security bulletin, because we know that no software development process will catch all security bugs. As we have said many times, the goal of the SDL is to "Reduce vulnerabilities, and reduce the severity of what's missed."

In this post, I want to focus on the SDL-required code analysis, code review, fuzzing and compiler and operating system defenses and how they fared.

Background

The bug was an invalid pointer dereference in MSHTML.DLL when the code handles data binding. It's important to point out that there is no heap corruption and there is no heap-based buffer overrun!

When data binding is used, IE creates an object which contains an array of data binding objects. In the code in question, when a data binding object is released, the array length is not correctly updated leading to a function call into freed memory.

The vulnerable code looks a little like this (by the way, the real array name is _aryPXfer, but I figured ArrayOfObjectsFromIE is a little more descriptive for people not in the Internet Explorer team.)

```
int MaxIdx = ArrayOfObjectsFromIE.Size()-1;
for (int i=0; i <= MaxIdx; i++) {
    if (!ArrayOfObjectsFromIE[i])
        continue;
    ArrayOfObjectsFromIE[i]->TransferFromSource();
    ...
}
```

Here's how the vulnerability manifests itself: if there are two data transfers with the same identifier (so MaxIdx is 2), and the first transfer updates the length of the ArrayOfObjectsFromIE array when its work was done and releases its data binding object, the loop count would still be whatever MaxIdx was at the start of the loop, 2.

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

The fix was to check the maximum iteration count on each loop iteration rather than once before the loop starts; this is the correct fix for a TOCTOU bug - move the check as close as possible to the action because, in

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.

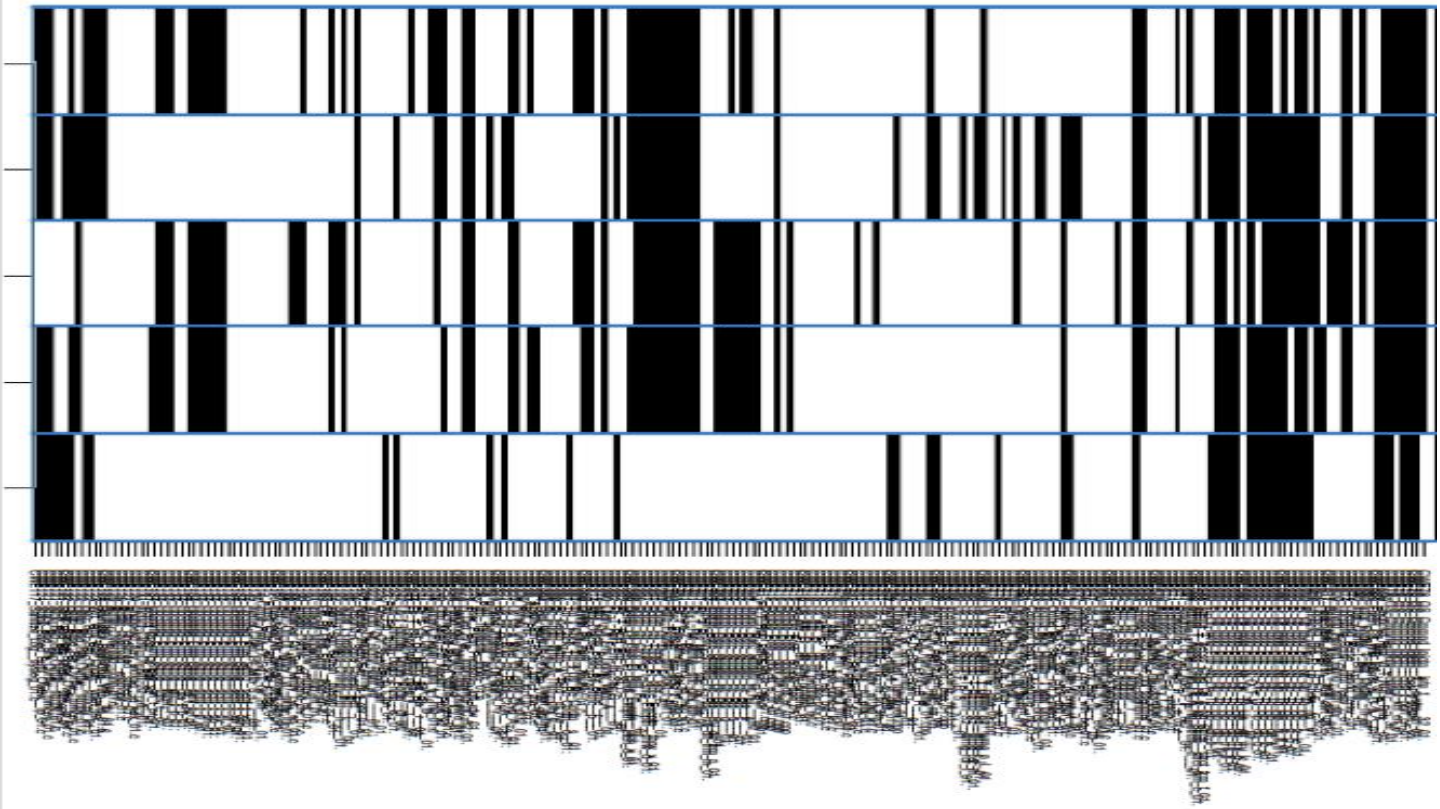
September 2008 (5)
August 2008 (2)
July 2008 (8)
June 2008 (4)

TOCTOU issues. We will update our training to address this.

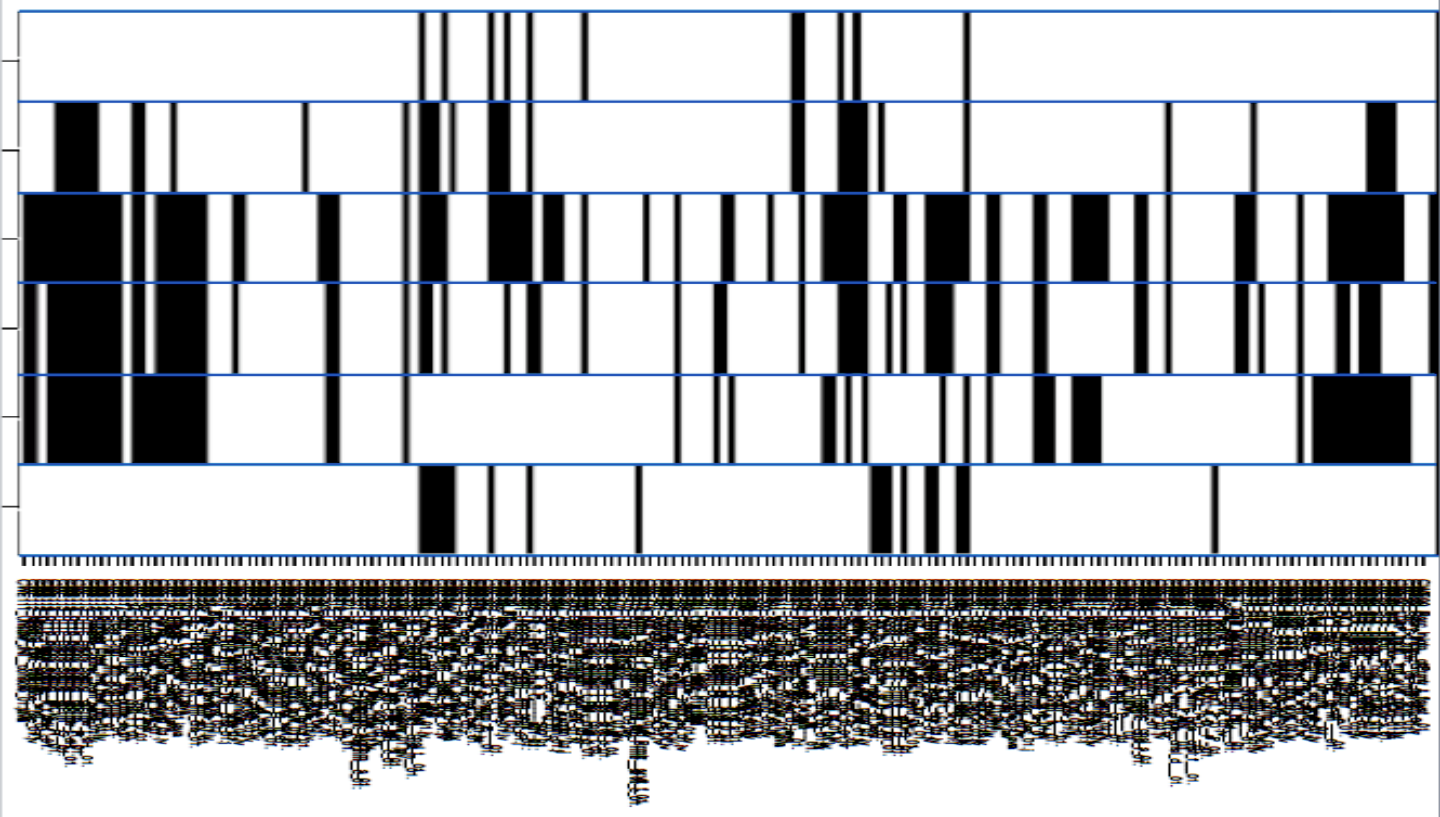
Our static analysis tools don't find this because the tools would need to understand the re-entrant nature of the code.

Fuzz Testing

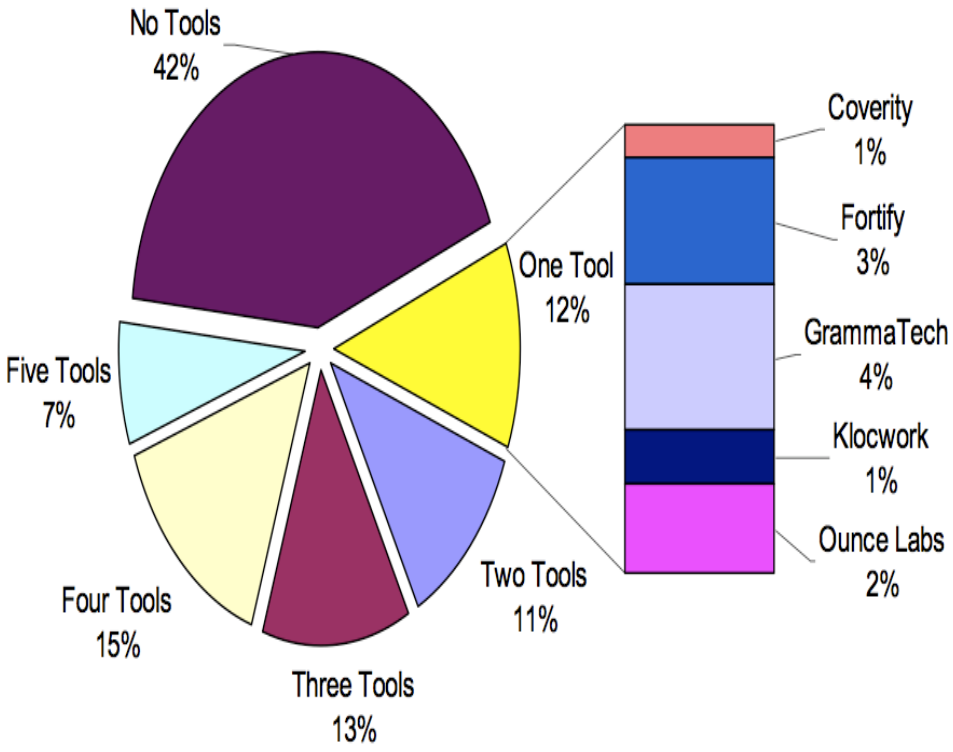
C Test Cases



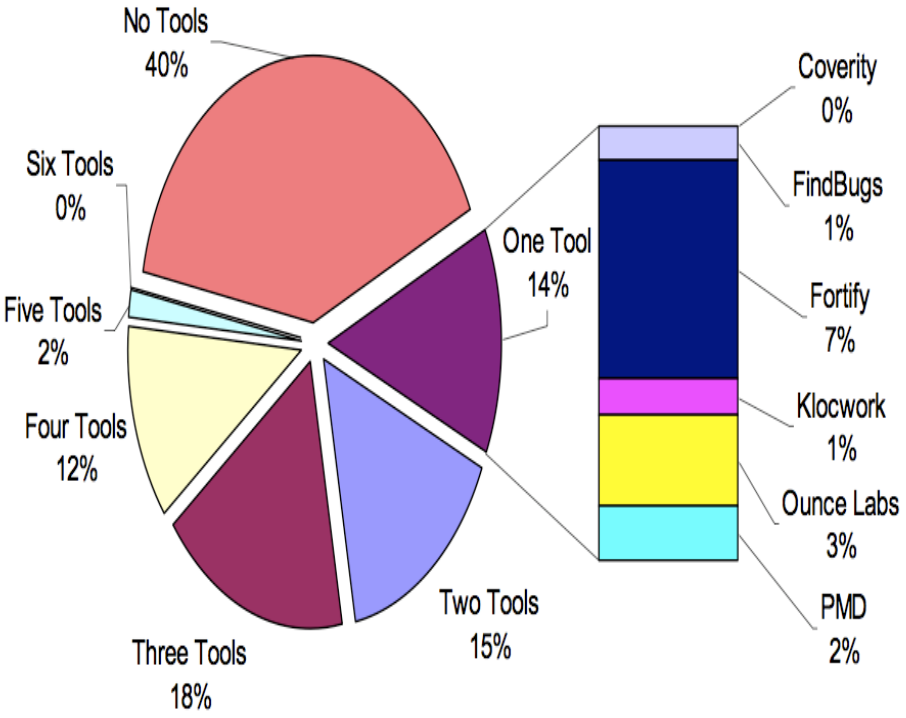
Java Test Cases



C/C++ "Breadth" Test Case Coverage



Java "Breadth" Test Case Coverage



MITRE

CWE/SANS Top 25 Programming Errors

- **Sponsored by:**
  - National Cyber Security Division (DHS)
  - Information Assurance Division (NSA)
- **List was selected by a group of security experts from 35 organizations including:**
  - Academia: Purdue, Univ. of Cal., N. Kentucky Univ.
  - Government: CERT, NSA, DHS
  - Software Vendors: Microsoft, Oracle, Red Hat, Apple
  - Security Vendors: Veracode, Fortify, Cigital, Symantec
- **Released in 2009, updated in 2010 and 2011**
- **Future versions possible**

Robert C. Seacord	CERT
Pascal Meunier	CERIAS, Purdue University
Matt Bishop	University of California, Davis
Kenneth van Wyk	KRvW Associates
Masato Terada	Information-Technology Promotion Agency (IPA) (Japan)
Sean Barnum	Cigital, Inc.
Mahesh Saptarshi	Symantec Corporation
Cassio Goldschmidt	Symantec Corporation
Adam Hahn	MITRE
Jeff Williams	Aspect Security and OWASP
Carsten Eiram	Secunia
Josh Drake	iDefense Labs at VeriSign, Inc.
Chuck Willis	MANDIANT
Michael Howard	Microsoft
Bruce Lowenthal	Oracle Corporation
Mark J. Cox	Red Hat Inc.
Jacob West	Fortify Software
Djenana Campara	Hatha Systems
James Walden	Northern Kentucky University
Frank Kim	ThinkSec
Chris Eng	Veracode, Inc.
Chris Wysopal	Veracode, Inc.
Ryan Barnett	Breach Security
Antonio Fontes	New Access SA (Switzerland)
Mark Fioravanti II	Missing Link Security Inc.
Ketan Vyas	Tata Consultancy Services (TCS)
Lindsey Cheng	Secured Sciences Group, LLC
Ian Peters	Secured Sciences Group, LLC
Tom Burgess	Secured Sciences Group, LLC
Hardik Parekh	RSA - Security Division of EMC Corporation
Matthew Coles	RSA - Security Division of EMC Corporation
Mouse	
Ivan Ristic	
Apple Product Security	
Software Assurance Forum for Excellence in Code (SAFECode)	
Core Security Technologies Inc.	
Depository Trust & Clearing Corporation (DTCC)	
The working group at the first OWASP ESAPI Summit	
National Security Agency (NSA) Information Assurance Division	
Department of Homeland Security (DHS) National Cyber Security Division	

Special thanks to Alan Paller and Mason Brown (SANS), and Janis Kenderdine and Conor Harris (MITRE)

MITRE

Main Goals of the Top 25

- Raise awareness for developers just starting out in security
- Help universities to teach secure coding
- Empower customers who want to ask for more secure software
- Provide a starting point for in-house software shops to measure their own progress

Rank	Score	ID	Name
[1]	93.8	CWE-89	Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection')
[2]	83.3	CWE-78	Improper Neutralization of Special Elements used in an OS Command ('OS Command Injection')
[3]	79.0	CWE-120	Buffer Copy without Checking Size of Input ('Classic Buffer Overflow')
[4]	77.7	CWE-79	Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')
[5]	76.9	CWE-306	Missing Authentication for Critical Function
[6]	76.8	CWE-862	Missing Authorization
[7]	75.0	CWE-798	Use of Hard-coded Credentials
[8]	75.0	CWE-311	Missing Encryption of Sensitive Data
[9]	74.0	CWE-434	Unrestricted Upload of File with Dangerous Type
[10]	73.8	CWE-807	Reliance on Untrusted Inputs in a Security Decision
[11]	73.1	CWE-250	Execution with Unnecessary Privileges
[12]	70.1	CWE-352	Cross-Site Request Forgery (CSRF)
[13]	69.3	CWE-22	Improper Limitation of a Pathname to a Restricted Directory ('Path Traversal')
[14]	68.5	CWE-494	Download of Code Without Integrity Check
[15]	67.8	CWE-863	Incorrect Authorization
[16]	66.0	CWE-829	Inclusion of Functionality from Untrusted Control Sphere
[17]	65.5	CWE-732	Incorrect Permission Assignment for Critical Resource
[18]	64.6	CWE-676	Use of Potentially Dangerous Function
[19]	64.1	CWE-327	Use of a Broken or Risky Cryptographic Algorithm
[20]	62.4	CWE-131	Incorrect Calculation of Buffer Size
[21]	61.5	CWE-307	Improper Restriction of Excessive Authentication Attempts
[22]	61.1	CWE-601	URL Redirection to Untrusted Site ('Open Redirect')
[23]	61.0	CWE-134	Uncontrolled Format String
[24]	60.3	CWE-190	Integer Overflow or Wraparound
[25]	59.9	CWE-759	Use of a One-Way Hash without a Salt

- **Insecure Interaction Between Components**
- **Risky Resource Management**
- **Porous Defenses**

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

MITRE

Recent Posts

SDL Threat Modeling Tool 3.1.4 ships!
Early Days of the SDL, Part Four
Early Days of the SDL, Part Three
Early Days of the SDL, Part Two
Early Days of the SDL, Part One

Tags

Common Criteria **Crawl Walk Run** Privacy **SDL** SDL Pro Network Security Assurance Security Blackhat SDL **threat modeling**

News

About Us

Adam Shostack
Bryan Sullivan
David Ladd
Jeremy Dallman
Michael Howard
Steve Lipner

Blogroll

BlueHat Security Briefings

SDL and the CWE/SANS Top 25

Bryan here. The security community has been buzzing since SANS and MITRE's joint announcement earlier this month of their list of the Top 25 Most Dangerous Programming Errors. Now, I don't want to get into a debate in this blog about whether this new list will become the new de facto standard for analyzing security vulnerabilities (or indeed, whether it already has become the new standard). Instead, I'd like to present an overview of how the Microsoft SDL maps to the CWE/SANS list, just
May.

Michael and I have writte
coverage of the Top 25 ar
believe that the results te
25 were developed indep
root them out of the softw
analysis white paper and
guidance around every m
made many of the same S
for you to download and u

Below is a summary of ho
see the SDL covers every
them (race conditions and
by multiple SDL requirem
tools to prevent or detect

CWE	Title
20	Improper Input Va
116	Improper Encodin Escaping of Outpu

CWE	Title	Education?	Manual Process?	Tools?	Threat Model?
20	Improper Input Validation	Y	Y	Y	Y
116	Improper Encoding or Escaping of Output	Y	Y	Y	
89	Failure to Preserve SQL Query Structure (aka SQL Injection)	Y	Y	Y	
79	Failure to Preserve Web Page Structure (aka Cross-Site Scripting)	Y	Y	Y	
78	Failure to Preserve OS Command Structure (aka OS Command Injection)	Y		Y	
319	Cleartext Transmission of Sensitive Information	Y			Y
352	Cross-site Request Forgery (aka CSRF)	Y		Y	
362	Race Condition	Y			
209	Error Message Information Leak	Y	Y	Y	
119	Failure to Constrain Memory Operations within the Bounds of a Memory Buffer	Y	Y	Y	
642	External Control of Critical State Data	Y			Y
73	External Control of File Name or Path	Y	Y	Y	
426	Untrusted Search Path	Y		Y	
94	Failure to Control Generation of Code (aka 'Code Injection')	Y	Y		
494	Download of Code Without Integrity Check				Y
404	Improper Resource Shutdown or Release	Y		Y	
665	Improper Initialization	Y		Y	
682	Incorrect Calculation	Y		Y	
285	Improper Access Control (Authorization)	Y	Y		Y
327	Use of a Broken or Risky Cryptographic Algorithm	Y	Y	Y	
259	Hard-Coded Password	Y	Y	Y	Y
732	Insecure Permission Assignment for Critical Resource	Y	Y		
330	Use of Insufficiently Random Values	Y	Y	Y	
250	Execution with Unnecessary Privileges	Y	Y		Y
602	Client-Side Enforcement of Server-Side Security	Y			Y

What are the Attacks that would be Effective Against Your Weaknesses?

1 **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. In 2011, SQL injection was responsible for the compromises of many high-profile organizations, including Sony Pictures, PBS, MySQL.com, security company HBGary Federal, and many others.

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 suc

Architecture and Design
If available, use structured mechanisms that autom
validation automatically, instead of relying on the d
Process SQL queries using prepared statements, pa
dynamically construct and execute query strings wi

Architecture and Design, Operation
Run your code using the lowest privileges that are
That way, a successful attack will not immediately
administrator, especially in day-to-day operations.
Specifically, follow the principle of least privilege wh
the requirements of the system indicate that a use
on all database objects, such as execute-only for st

Architecture and Design
For any security checks that are performed on the
by modifying values after the checks have been pe

Implementation
If you need to use dynamically-generated query str
conservative approach is to escape or filter all char
are still needed, such as white space, wrap each ar
Instead of building your own implementation, such
that parameters have certain properties that make

Implementation
Assume all input is malicious. Use an "accept know

Implementation
Ensure that error messages only contain minimal details that are useful to the intended audience, and nobody else. The messages need to strike the balance between being too cryptic and not being cryptic enough. They should not necessarily reveal the methods that were used to determine the error. Such detailed information can be used to refine the original attack to increase the chances of success.
If errors must be tracked in some detail, capture them in log messages - but consider what could occur if the log messages can be viewed by attackers. Avoid recording highly sensitive information such as passwords in any form. Avoid inconsistent messaging that might accidentally tip off an attacker about internal state, such as whether a username is valid or not.

In the context of SQL Injection, error messages revealing the structure of a SQL query can help attackers tailor successful attack strings.

Operation
Use an application firewall that can detect attacks against this weakness. It can be beneficial in cases in which the code cannot be fixed (because it is controlled by a third party), as an emergency prevention measure while more comprehensive software assurance measures are applied, or to provide defense in depth.
Effectiveness: Moderate

Notes: An application firewall might not cover all possible input vectors. In addition, attack techniques might be available to bypass the protection mechanism, such as using malformed inputs that can still be processed by the component that receives those inputs. Depending on functionality, an application firewall might inadvertently reject or modify legitimate requests. Finally, some manual effort may be required for customization.

Operation, Implementation
If you are using PHP, configure your application so that it does not use register_globals. During implementation, develop your application so that it does not rely on this feature, but be wary of implementing a register_globals emulation that is subject to weaknesses such as CWE-95, CWE-621, and similar issues.

Related CWEs

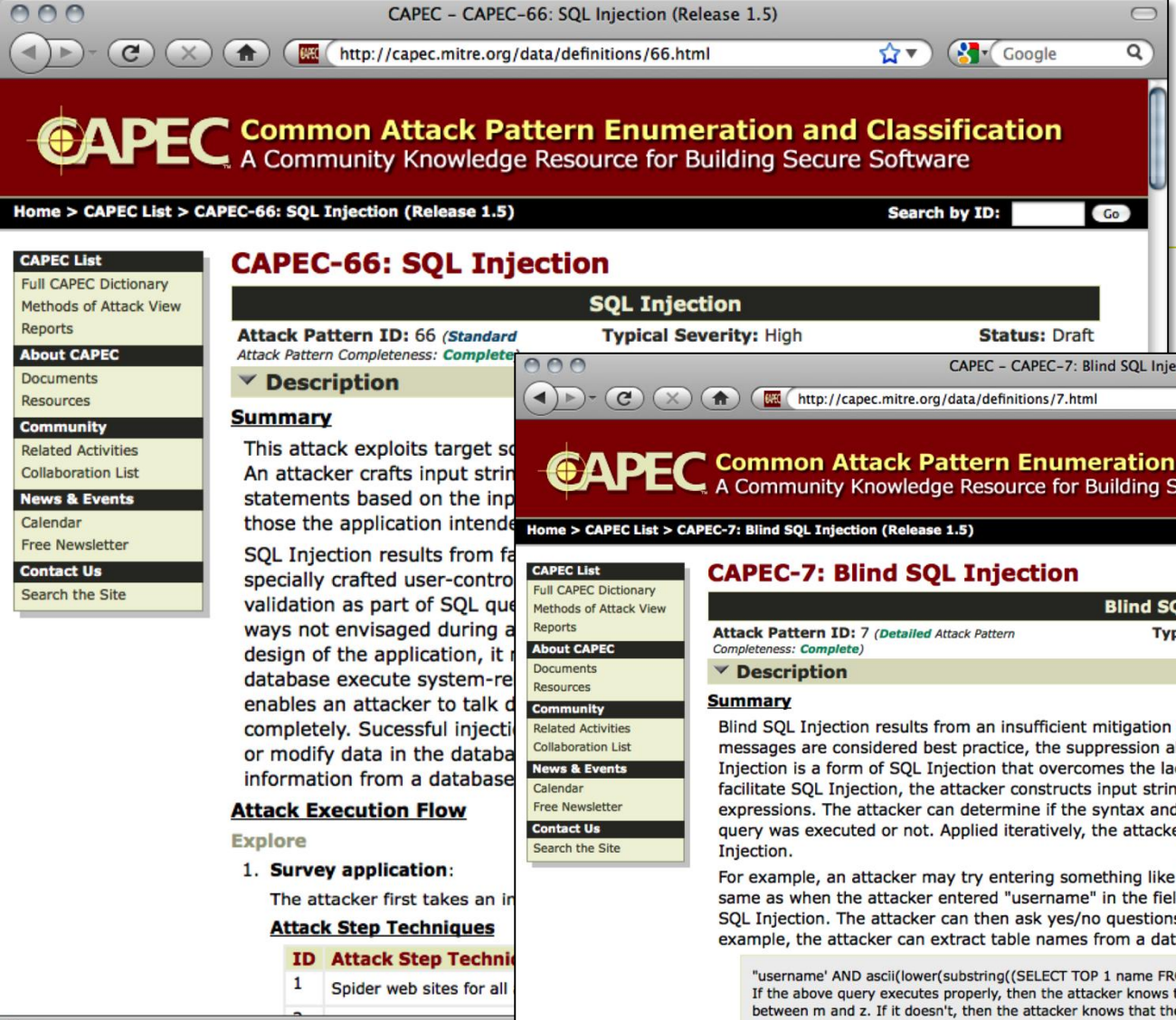
CWE-90	Improper Neutralization of Special Elements used in an LDAP Query ('LDAP Injection')
CWE-564	SQL Injection: Hibernate
CWE-566	Authorization Bypass Through User-Controlled SQL Primary Key
CWE-619	Dangling Database Cursor ('Cursor Injection')

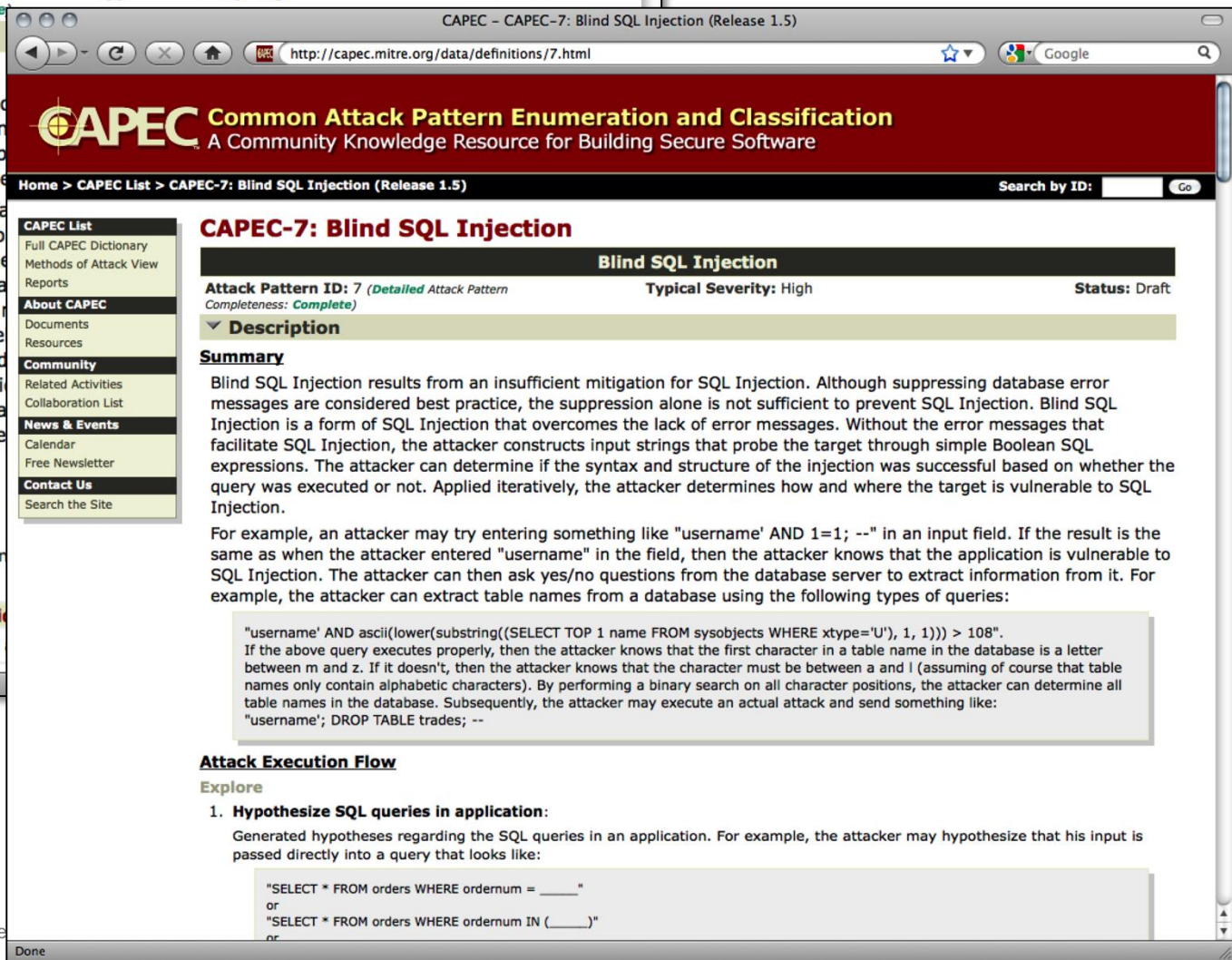
Related Attack Patterns

CAPEC-IDs: [view all]
7, 66, 108, 109, 110

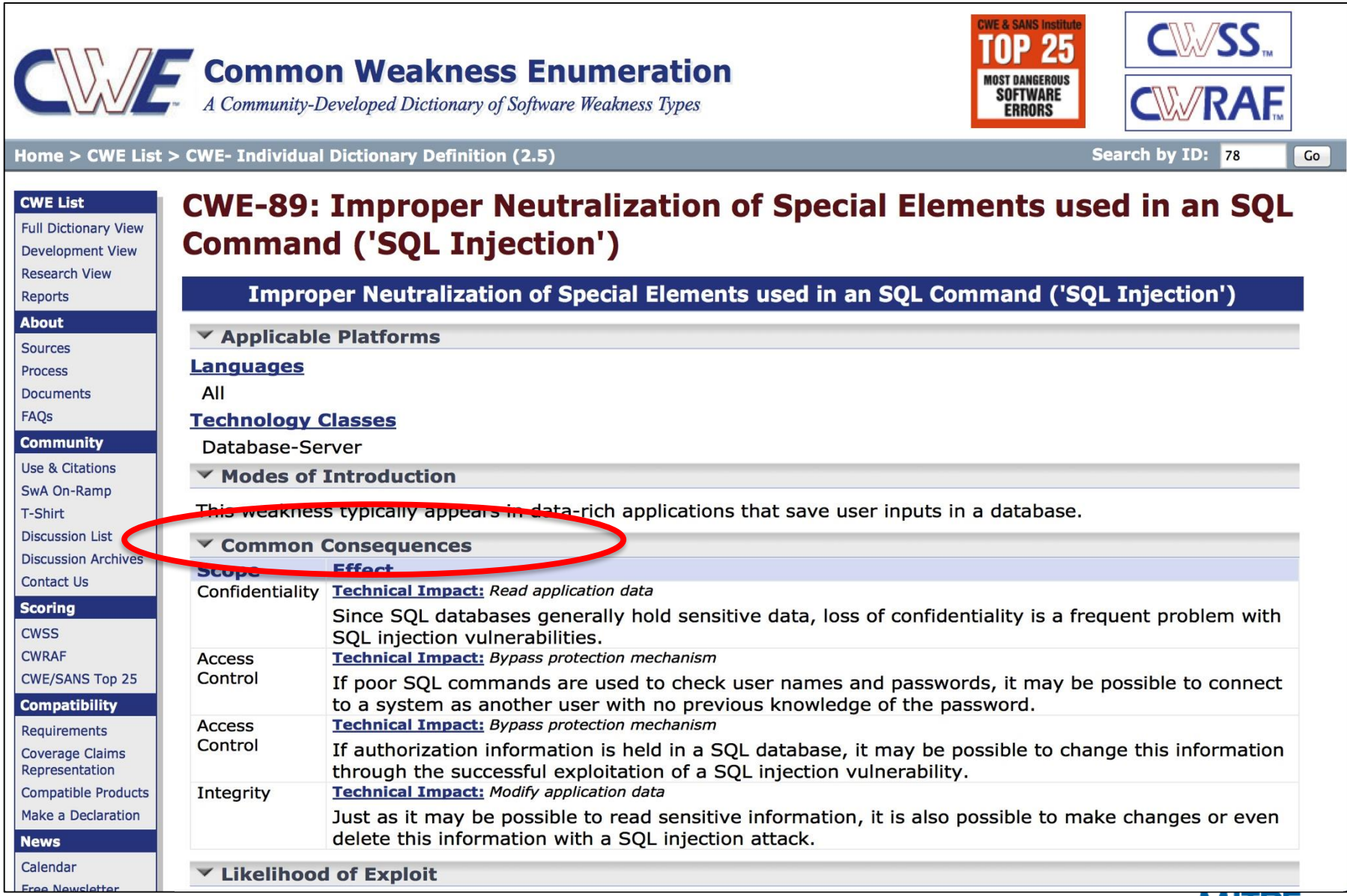
MITRE

http://capec.mitre.org/





Prioritizing by Technical Impacts: CWE's Common Consequences



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

CWE & SANS Institute TOP 25 MOST DANGEROUS SOFTWARE ERRORS

CWSS

CWRAF

Home > CWE List > CWE- Individual Dictionary Definition (2.5)

Search by ID: 78 Go

CWE List
Full Dictionary View
Development View
Research View
Reports
About
Sources
Process
Documents
FAQs
Community
Use & Citations
SwA On-Ramp
T-Shirt
Discussion List
Discussion Archives
Contact Us
Scoring
CWSS
CWRAF
CWE/SANS Top 25
Compatibility
Requirements
Coverage Claims Representation
Compatible Products
Make a Declaration
News
Calendar
Free Newsletter

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')

Applicable Platforms

Languages

All

Technology Classes

Database-Server

Modes of Introduction

This weakness typically appears in data-rich applications that save user inputs in a database.

Common Consequences

Scope	Effect
Confidentiality	**Technical Impact:** *Read application data* Since SQL databases generally hold sensitive data, loss of confidentiality is a frequent problem with SQL injection vulnerabilities.
Access Control	**Technical Impact:** *Bypass protection mechanism* If poor SQL commands are used to check user names and passwords, it may be possible to connect to a system as another user with no previous knowledge of the password.
Access Control	**Technical Impact:** *Bypass protection mechanism* If authorization information is held in a SQL database, it may be possible to change this information through the successful exploitation of a SQL injection vulnerability.
Integrity	**Technical Impact:** *Modify application data* Just as it may be possible to read sensitive information, it is also possible to make changes or even delete this information with a SQL injection attack.

Likelihood of Exploit

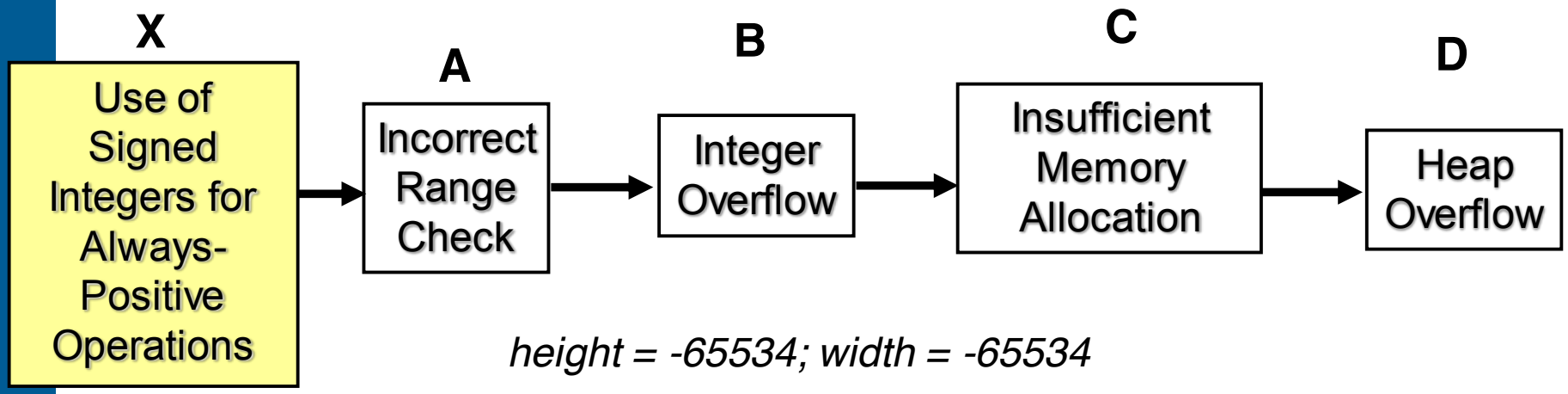
Technical Impact	Automated Analysis	Automated Dynamic Analysis	Automated Static Analysis	Black Box	Fuzzing	Manual Analysis	Manual Dynamic Analysis	Manual Static Analysis	White Box
Execute unauthorized code or commands		78, 120, 129, 131, 476, 805	78, 79, 98, 120, 129, 131, 134, 190, 426, 798, 805	79, 129, 134, 190, 426, 494, 698, 798		98, 120, 131, 190, 426, 494, 805	476, 798	78, 798	
Gain privileges / assume identity		601	306, 352, 426, 601, 798	259, 426, 798		259, 306, 352, 426	798	601, 798, 807	
Read data	209, 311, 327	78, 89, 129, 131, 209, 404, 665	78, 79, 89, 129, 131, 134, 352, 426, 798	14, 79, 129, 134, 319, 426, 798		89, 131, 209, 311, 327, 352, 426	209, 404, 665, 798	78, 798	14
Modify data	311, 327	78, 89, 129, 131	78, 89, 129, 131, 190, 352	129, 190, 319		89, 131, 190, 311, 327, 352		78	
DoS: unreliable execution		78, 120, 129, 131, 400, 476, 665, 805	78, 120, 129, 131, 190, 352, 400, 426, 805	129, 190, 426, 690	400	120, 131, 190, 352, 426, 805	476, 665	78	
DoS: resource consumption		120, 400, 404, 770, 805	120, 190, 400, 770, 805	190	400, 770	120, 190, 805	404	770	412
Bypass protection mechanism		89, 400, 601, 665	79, 89, 190, 352, 400, 601, 798	14, 79, 184, 190, 733, 798	400	89, 190, 352	665, 798	601, 798, 807	14, 733
Hide activities	327	78	78			327		78	

Challenges and complexities…

or, why some vulnerabilities are still with us

MITRE

Chains: Why Buffer Overflows are Still Here

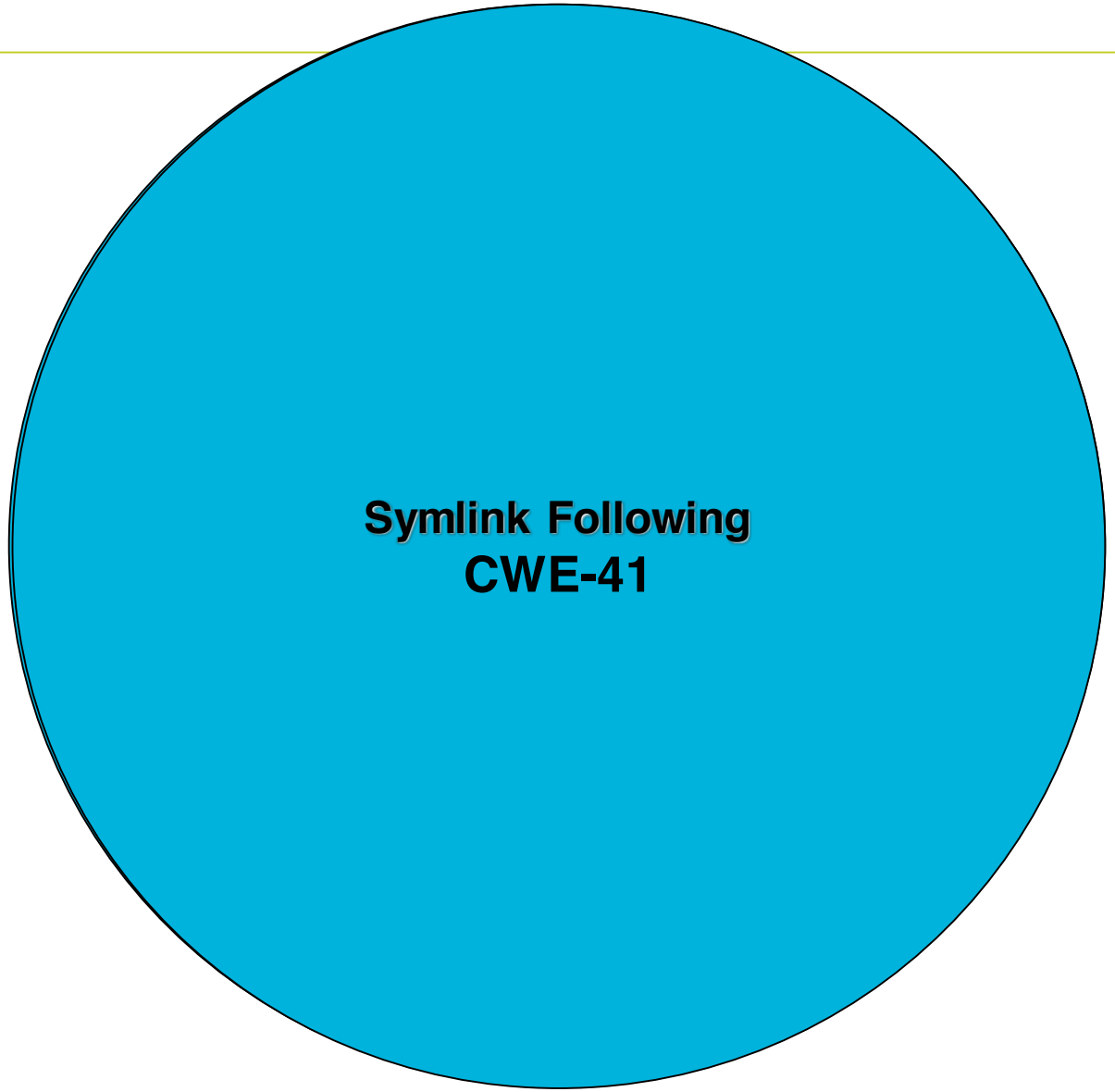


height = -65534; width = -65534

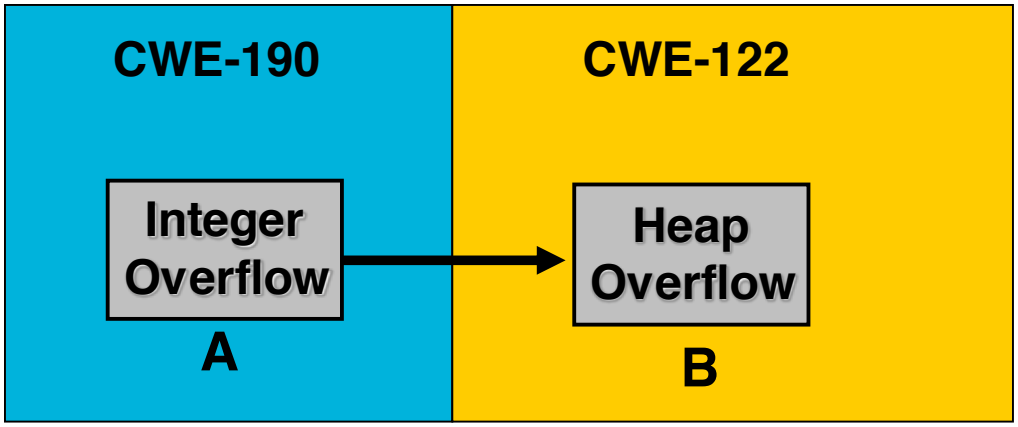
Assumption: the range check will prevent an overflow from occurring.

```
A  if (height > 64000 ||
       width > 64000) {
       error("too big!");
   }
B  size = height * width;
C  buf = malloc(size);
D  memmove(buf, InputBuf, SZ);
```

Symbolic Link Following



Named Chain Example: Integer Overflow to Heap Overflow (CWE-680)



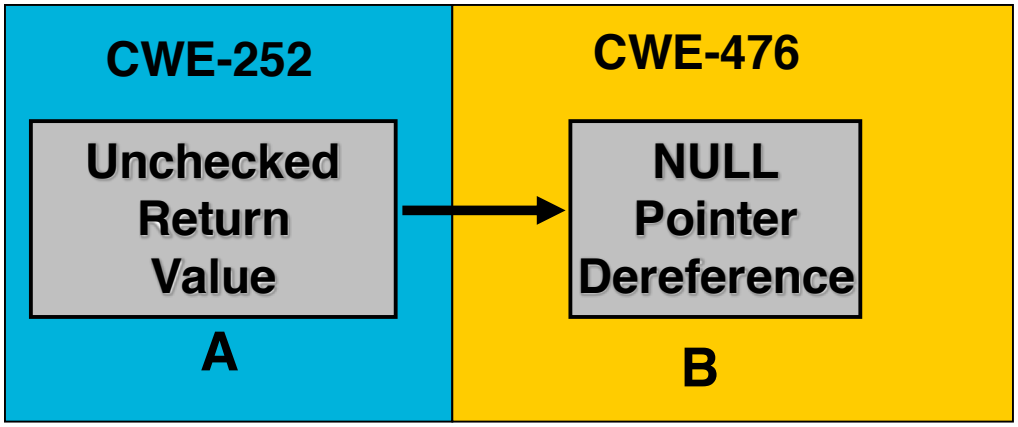
Assumption: height and width are reasonable sizes.

height = 65534; width = 65534

A
```
size = height * width;
buf = malloc(size);
```
B
```
memmove(buf, InputBuf, SZ);
```

The buffer overflow occurs because the newly created buffer is smaller than expected, because the integer overflow causes the 'size' variable to be smaller than expected.

Named Chain Example: Unchecked Return Value to NULL Pointer Dereference (CWE-690)



Assumption: height and width are reasonable sizes.

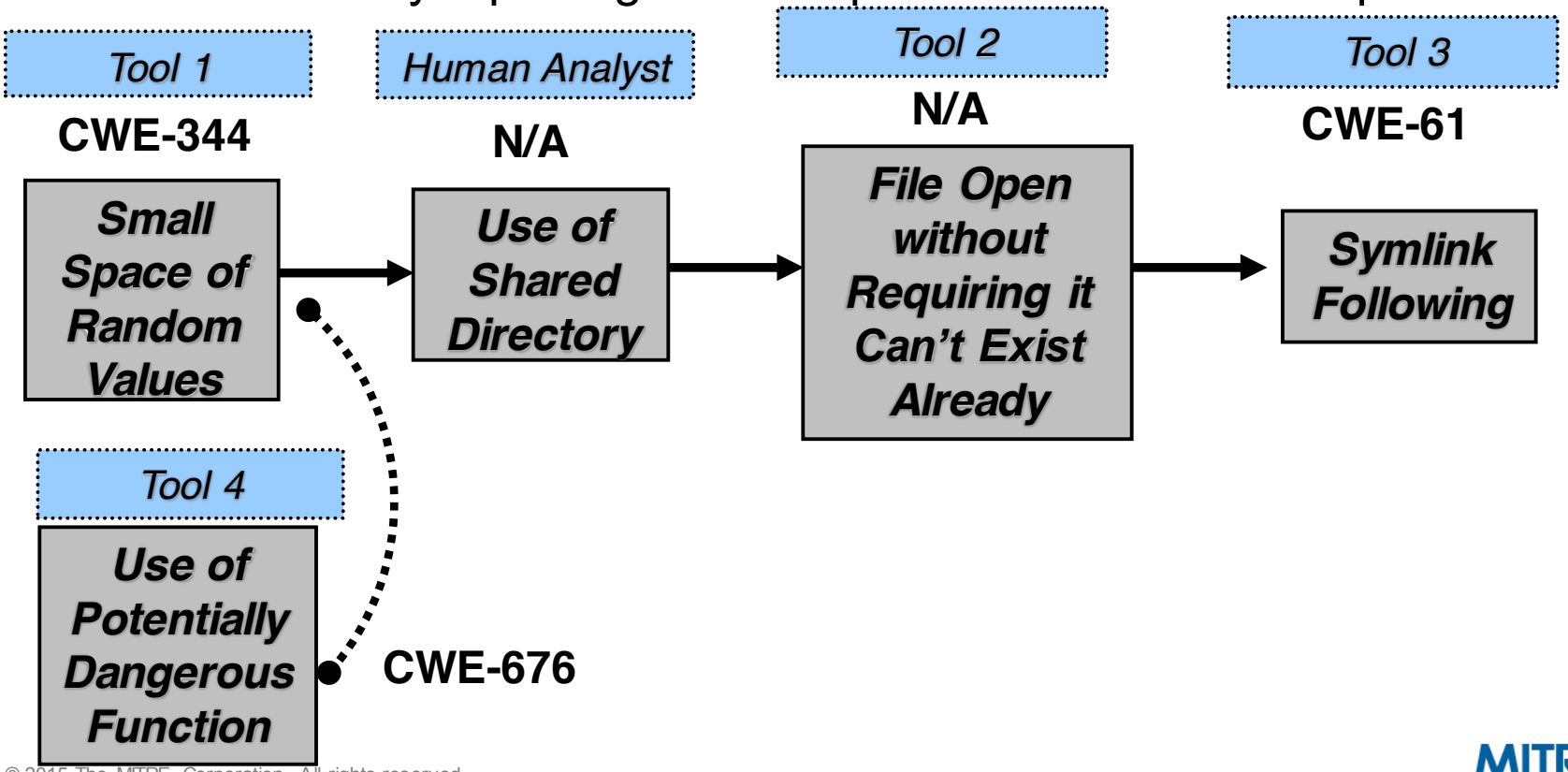
height = 63000; width = 63000

```
A  size = height * width;
   buf = malloc(size);
B  memmove(buf, InputBuf, SZ);
```

With properly selected height and width, an extremely large size value could cause malloc to return NULL due to out-of-memory conditions.

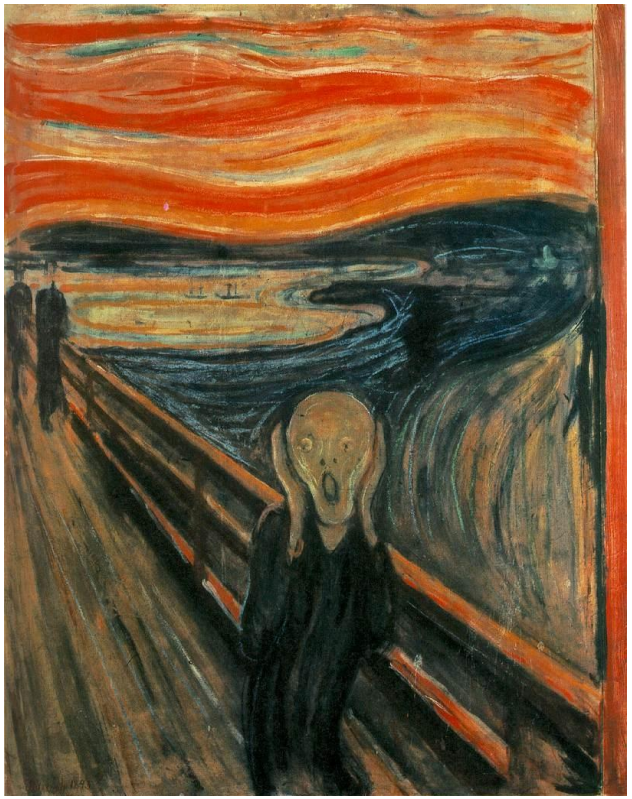
Chains, Composites, and Code Scanning

- **Comparisons between code scanning capabilities can yield significantly different results**
- **Very little overlap between tools**
  - … but are they reporting different parts of a chain or composite?



The Four I's Principle of Vulnerability Information

- **Incomplete**
  - Missing versions, product names
  - Missing patch information
- **Inaccurate**
  - Incorrect diagnosis
  - Blatantly wrong
- **Inconsistent**
  - Acknowledgement discrepancies
  - Bug type discrepancies
  - Varying severities
- **Incomprehensible**
  - Poor writing
  - Lack of clear formatting



Coordinated disclosure between researcher and vendor frequently wipes these out.

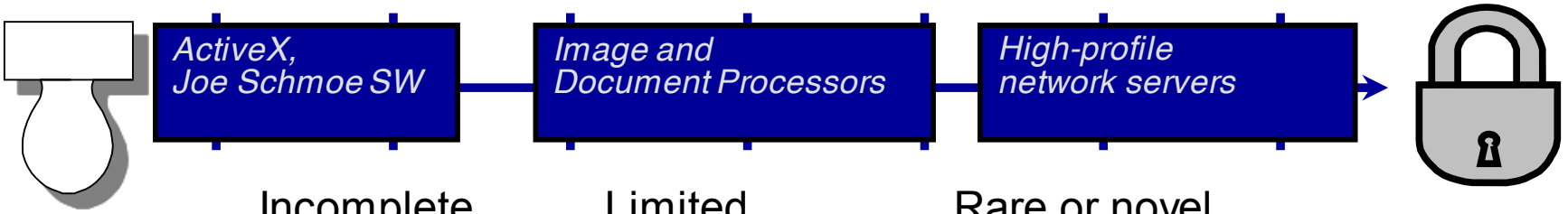
Typical Vulnerability History of a Product (Using 2007-era Examples)

1 Obvious types in critical functionality

3 Variants of common vulnerability types

5 Elimination of most common types

7 Unique types or attacks, extensive expert analysis



Incomplete fixes, closely related vectors

Limited environments, platforms, configs

Rare or novel types and attacks

2

4

6

MITRE

2007 Theories with 2015 Applications: Things. And Stuff.

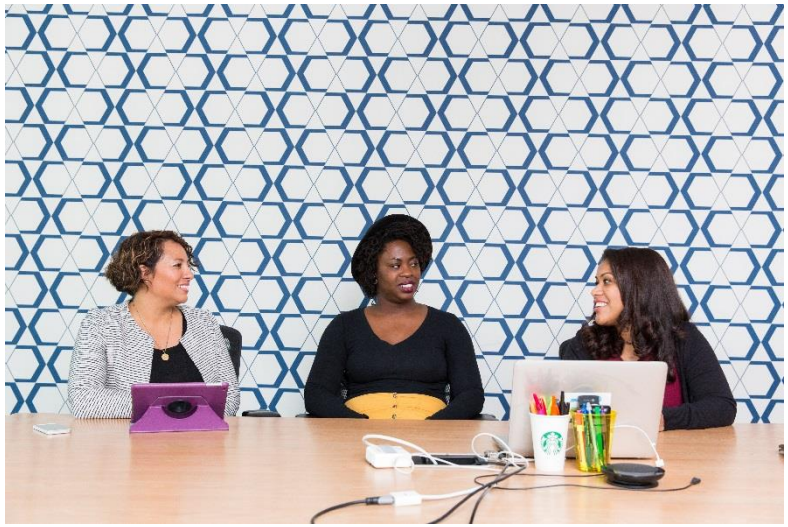
- Skateboard Bluetooth PIN guessing – property damage, human harm
- Toilet – environmental resource consumption, noise, "user inconvenience"
- Infusion pump, other medical devices – too much, too little, too late
- Coffee maker – electricity consumption, fire risk
- Voting machine – loss of confidentiality, integrity, availability, and democracy
- uConnect car entertainment system - human harm, loss of precious national resources such as Forbes journalists

Futures

- **CWE**
  - New weaknesses, variations of old themes
  - Greater emphasis on design-level and "new" product classes, e.g. mobile or medical
  - Engagement with academic community
    - Lots of potential for research!
- **CVE**
  - Low hanging fruit – gone?
  - Scale, scale, scale!
  - Automation / fuzzing
  - Massive influx of new/inexperienced researchers
  - Process changes to increase CVE output and make it more reliable
  - Focus on key products and data sources; no longer "all" vulnerabilities
- **Top 25**
  - Time for a new one?
  - But a general Top 25 can be of limited use
    - … yet we get asked about it ALL THE TIME
  - Ideal: customizable Top 25 lists
  - Next version (or variant) in 2015

Unsolicited Career Guidance

- **Self-directed career opportunities**
  - You can find a niche
  - Many careers don't require hard-core exploits
  - Local conferences are (relatively) cheap, and you can volunteer
- **Skillz**
  - Good writing and communication are extremely rare and extremely valuable
  - Empathy (for developers, users, peers, etc.) also extremely rare and (increasingly) valuable
  - Fundamental computing and networking is helpful but not necessary?
- **Educating yourself**
  - Try to know what you don't know
  - Learn the "mindset"
    - The older generation (i.e. me) aren't necessarily doing a good job of this
    - What separates a "bug" from a "vulnerability" from a "feature?"
  - Bug bounties



Thoughts? Questions? Answers?

Credit: #WOCinTechChat

Contact Me

@sushidude

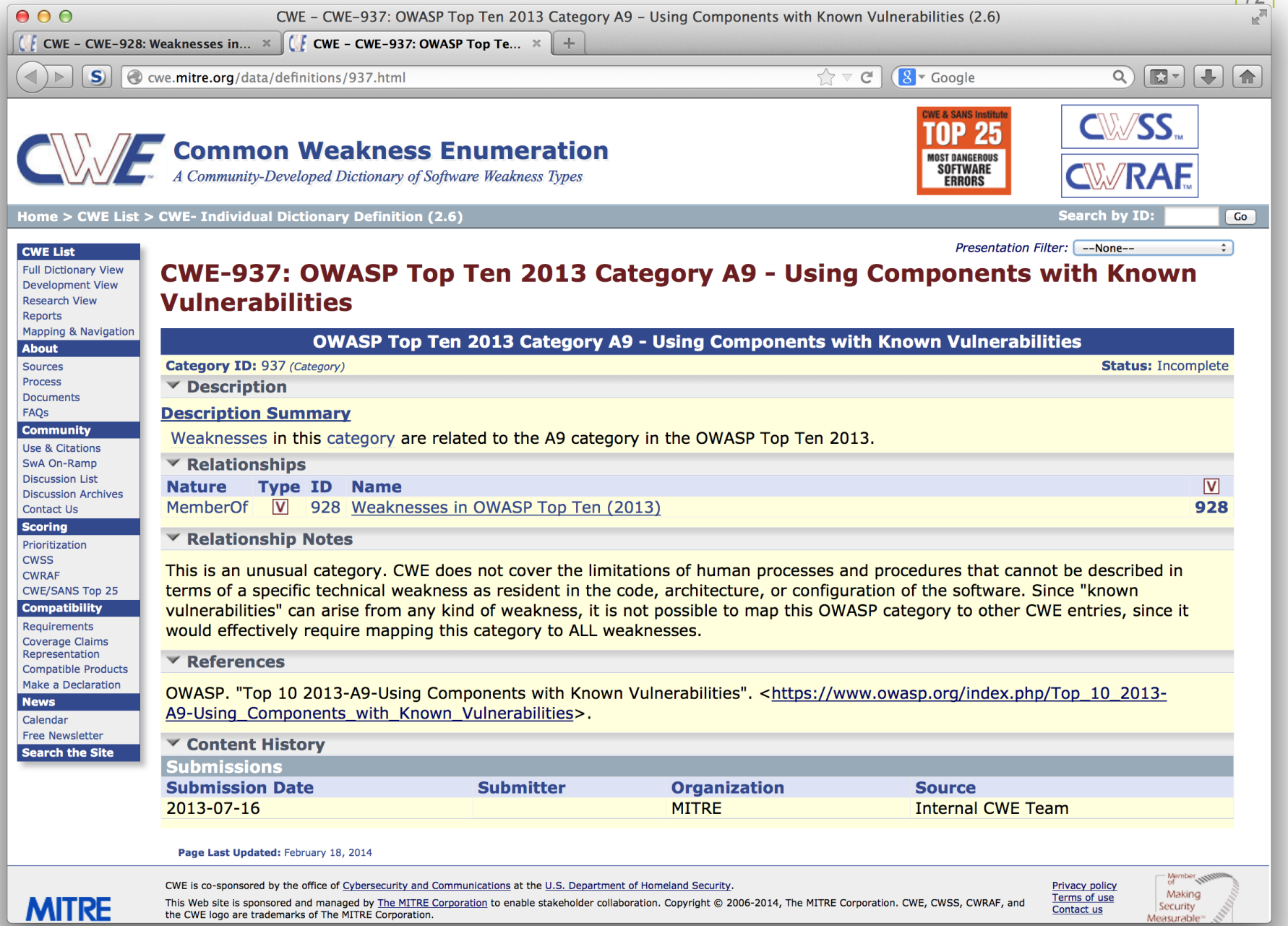
coley@mitre.org

Backup Slides

MITRE

The Classification Problem: Same Term, Many Perspectives, Lots of Overlap

Term	Attack	Vuln/Weakness	Consequence
Buffer Overflow	Long string argument Length field inconsistency Large number of events, etc.	Failure to restrict length Failure to control offset Error in attempting to do either of these	Write of data past explicitly specified boundaries of a buffer Crash, code execution, control/data flow modification
Format String	Format string specifiers relative to the underlying representation in use (typically C-style strings)	Failure to fully control contents of format strings	Write of data past explicitly specified boundaries of a buffer Crash, code execution, control/data flow modification
Directory Traversal	"..", "/a/b/c", "..../", etc.	Failure to properly restrict file within intended subdirectory	Access of file outside intended subdirectory
Information Leak	Provide invalid argument Monitor behavioral or timing results Sniff	Failure to anticipate error conditions Failure to limit info in error messages Failure to zero out sensitive info	Disclosure of sensitive information relative to an implicit or explicit policy of what constitutes "sensitive"
XSS	<SCRIPT>alert('hi')</SCRIPT> "javascript:alert(document.cookie)" "java#42;script:abc" …	Failure to properly filter, escape, or encode outputs with respect to their particular role (e.g. tags or tag arguments), in a fashion that is syntactically or semantically valid for the representation and encoding that are currently in use	Execution of script code Modification of format or presentation
DoS	Provide invalid argument	Failure to anticipate or handle error conditions Failure to properly limit scope of an error	Crash "Memory Corruption" Infinite loop
DoS	Large number of events Send a large amount of data Manipulate algorithmic complexity	Failure to sufficiently control resource consumption relative to performance expectations for the application and/or its environment	Crash "Memory Corruption" Infinite loop
Authentication Bypass	Perform invalid sequence of instructions, e.g. direct request Replay challenge/response Cookie modification SQL injection	Failure to enforce required sequence of steps Failure to prevent modification of assumed-immutable data Secondary effect of primary issue	Access privileged functionality or data before fully navigating all required authentication steps



CWE – CWE-937: OWASP Top Ten 2013 Category A9 – Using Components with Known Vulnerabilities (2.6)

CWE – CWE-928: Weaknesses in... | CWE – CWE-937: OWASP Top Te...

cwe.mitre.org/data/definitions/937.html

Google

Common Weakness Enumeration

A Community-Developed Dictionary of Software Weakness Types

CWE & SANS Institute TOP 25 MOST DANGEROUS SOFTWARE ERRORS

CWSS

CWRAF

Home > CWE List > CWE- Individual Dictionary Definition (2.6)

Search by ID: Go

CWE List
Full Dictionary View
Development View
Research View
Reports
Mapping & Navigation
About
Sources
Process
Documents
FAQs
Community
Use & Citations
SwA On-Ramp
Discussion List
Discussion Archives
Contact Us
Scoring
Prioritization
CWSS
CWRAF
CWE/SANS Top 25
Compatibility
Requirements
Coverage Claims Representation
Compatible Products
Make a Declaration
News
Calendar
Free Newsletter
Search the Site

Presentation Filter: --None--

CWE-937: OWASP Top Ten 2013 Category A9 - Using Components with Known Vulnerabilities

OWASP Top Ten 2013 Category A9 - Using Components with Known Vulnerabilities

Category ID: 937 *(Category)* **Status:** Incomplete

Description

Description Summary

Weaknesses in this category are related to the A9 category in the OWASP Top Ten 2013.

Relationships

Nature	Type	ID	Name	V
MemberOf	V	928	Weaknesses in OWASP Top Ten (2013)	928

Relationship Notes

This is an unusual category. CWE does not cover the limitations of human processes and procedures that cannot be described in terms of a specific technical weakness as resident in the code, architecture, or configuration of the software. Since "known vulnerabilities" can arise from any kind of weakness, it is not possible to map this OWASP category to other CWE entries, since it would effectively require mapping this category to ALL weaknesses.

References

OWASP. "Top 10 2013-A9-Using Components with Known Vulnerabilities". <https://www.owasp.org/index.php/Top_10_2013-A9-Using_Components_with_Known_Vulnerabilities>.

Content History

Submissions

Submission Date	Submitter	Organization	Source
2013-07-16		MITRE	Internal CWE Team

Page Last Updated: February 18, 2014

MITRE

CWE is co-sponsored by the office of Cybersecurity and Communications at the U.S. Department of Homeland Security.

This Web site is sponsored and managed by The MITRE Corporation to enable stakeholder collaboration. Copyright © 2006-2014, The MITRE Corporation. CWE, CWSS, CWRAF, and the CWE logo are trademarks of The MITRE Corporation.

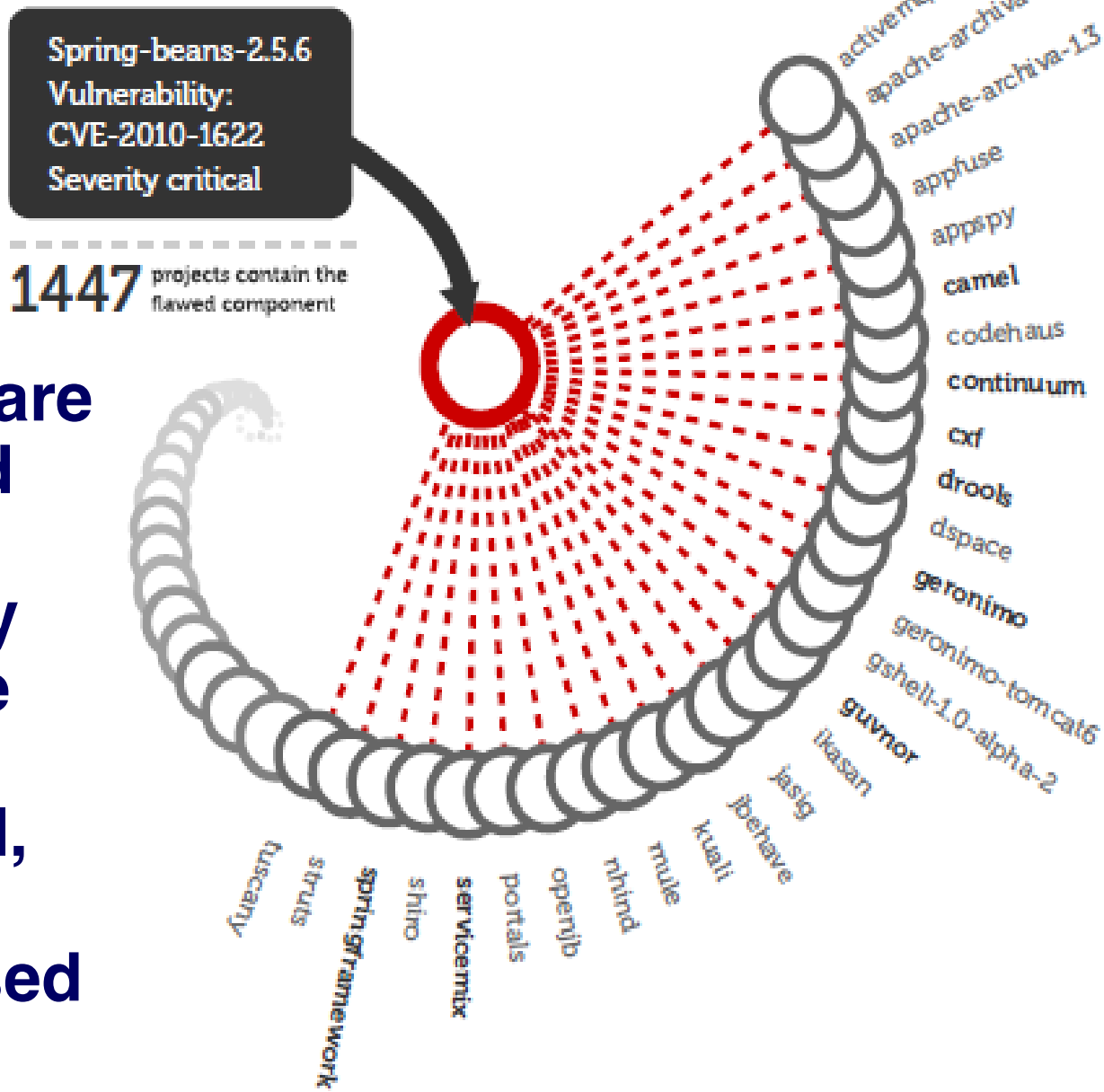
Privacy policy
Terms of use
Contact us

Member of Making Security Measurable

MITRE

CWE-937

Even after vulnerabilities are discovered and patches made available, many developers use (or continue to ues) the flawed, non-patched version of reused components



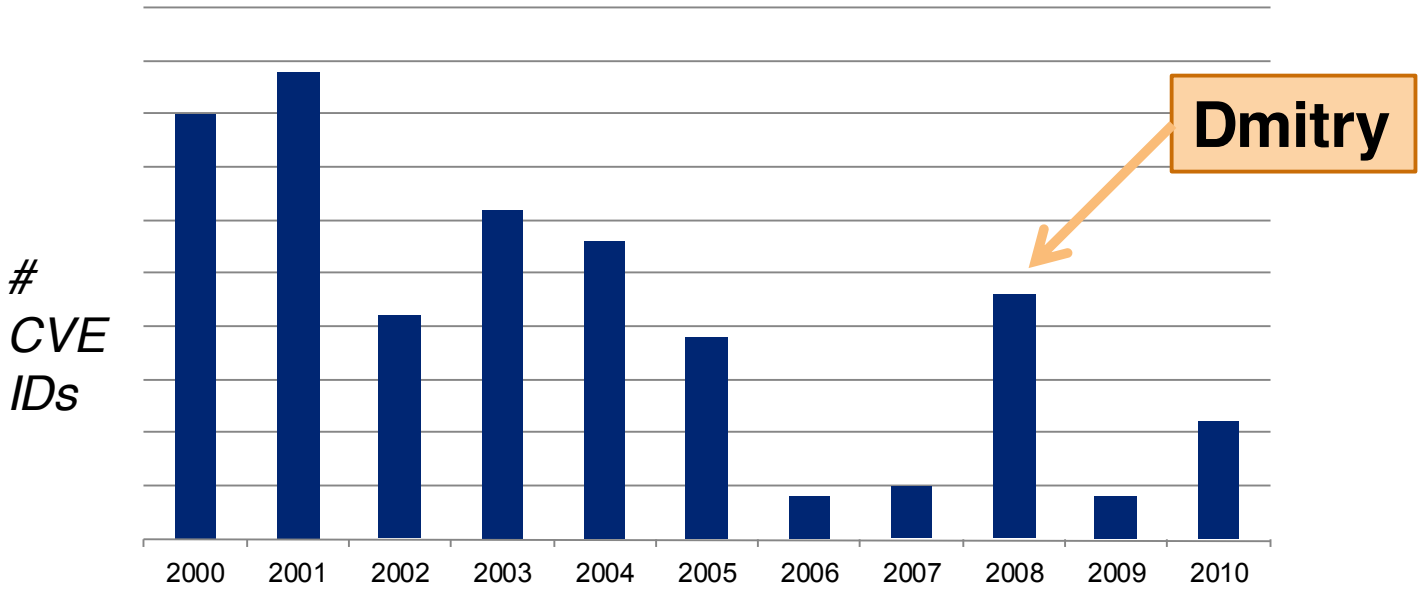
Source: Maximizing Benefits and Mitigating Risks of Open Source Components in Application Development, by Sonatype

MITRE

Grep-and-Gripe: Revenge of the Symlinks

```
grep -A5 -B5 /tmp/ $PROGRAM
```

- Dmitry E. Oboukhov, August 2008
- Run against Debian packages
- This kind of thing really hurts pie charts of different vulnerability types



Raw number of symlinks reported over time (CVE)

MITRE

Grep-and-Gripe 2: Larry Cashdollar*

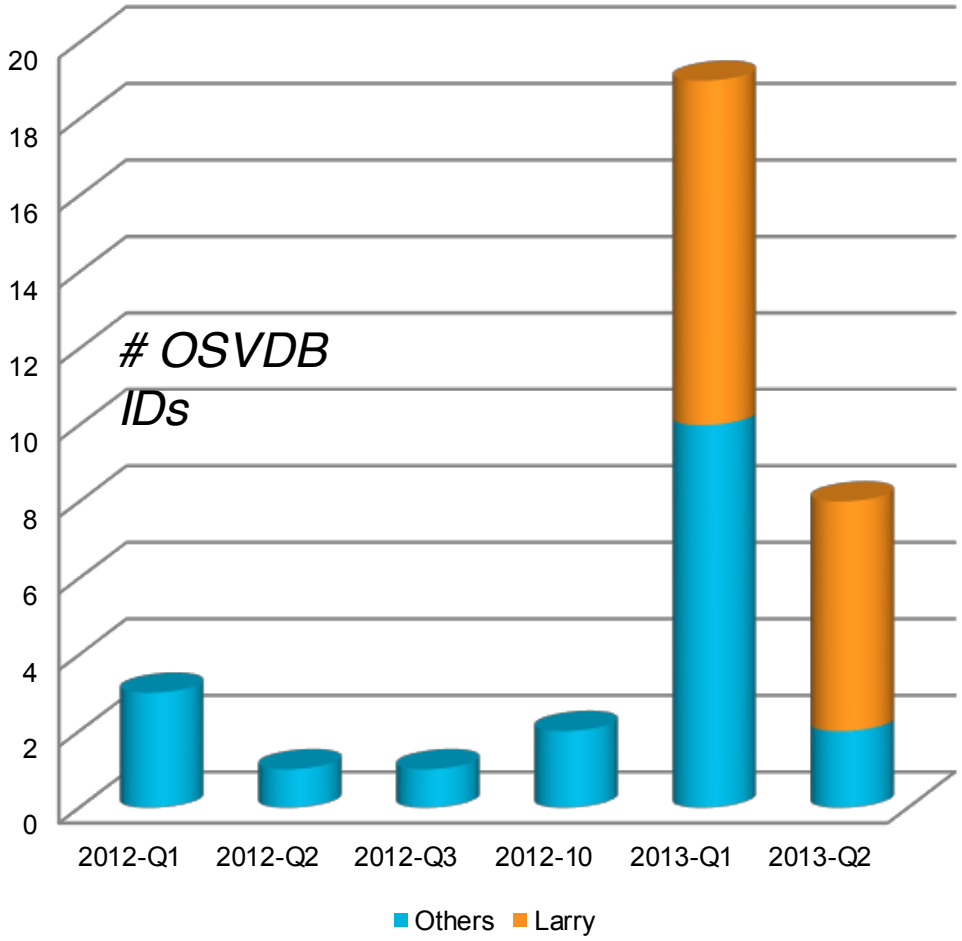
* That's his real last name. He swears it!

Grep-and-gripe

Old-school symbolic links and context-dependent OS command injection

Those are dead, right?

Enter Ruby Gems

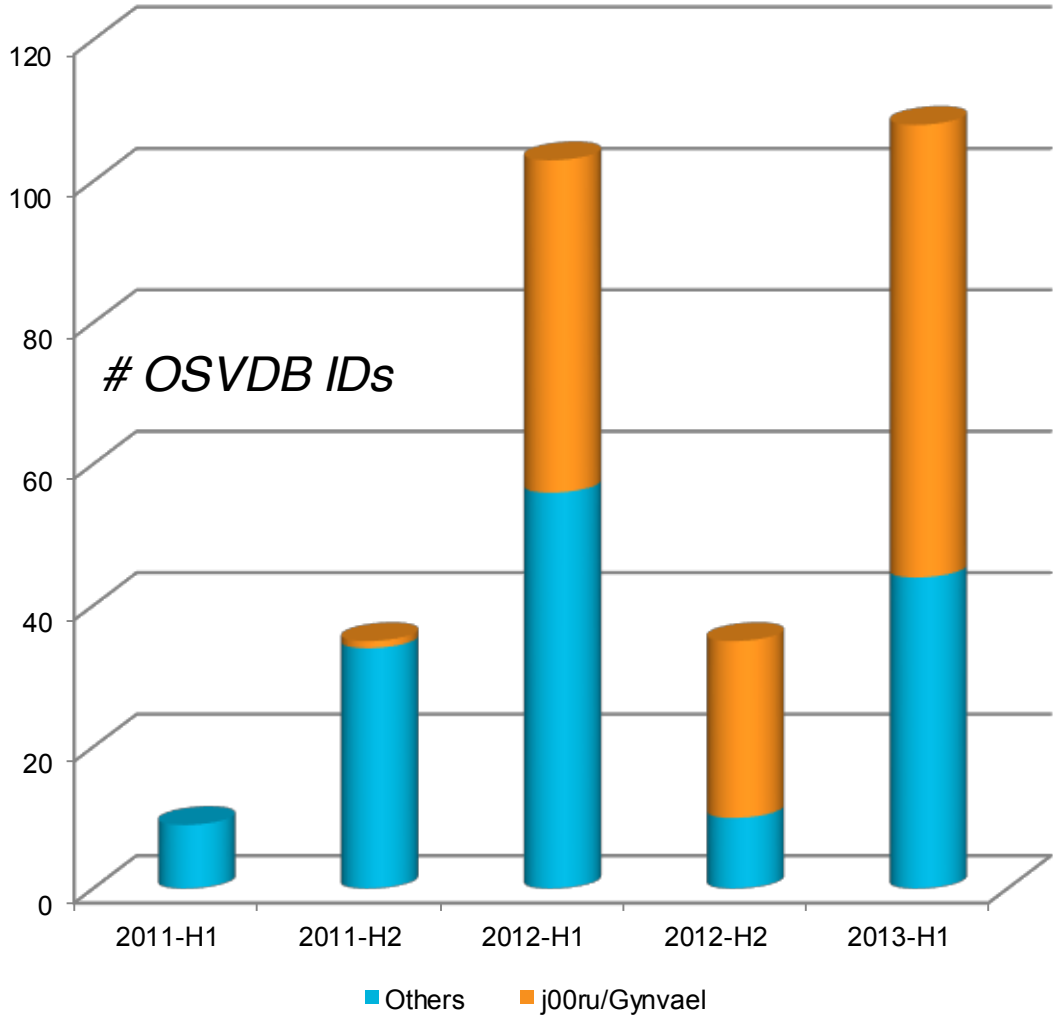


MITRE

FFmpeg

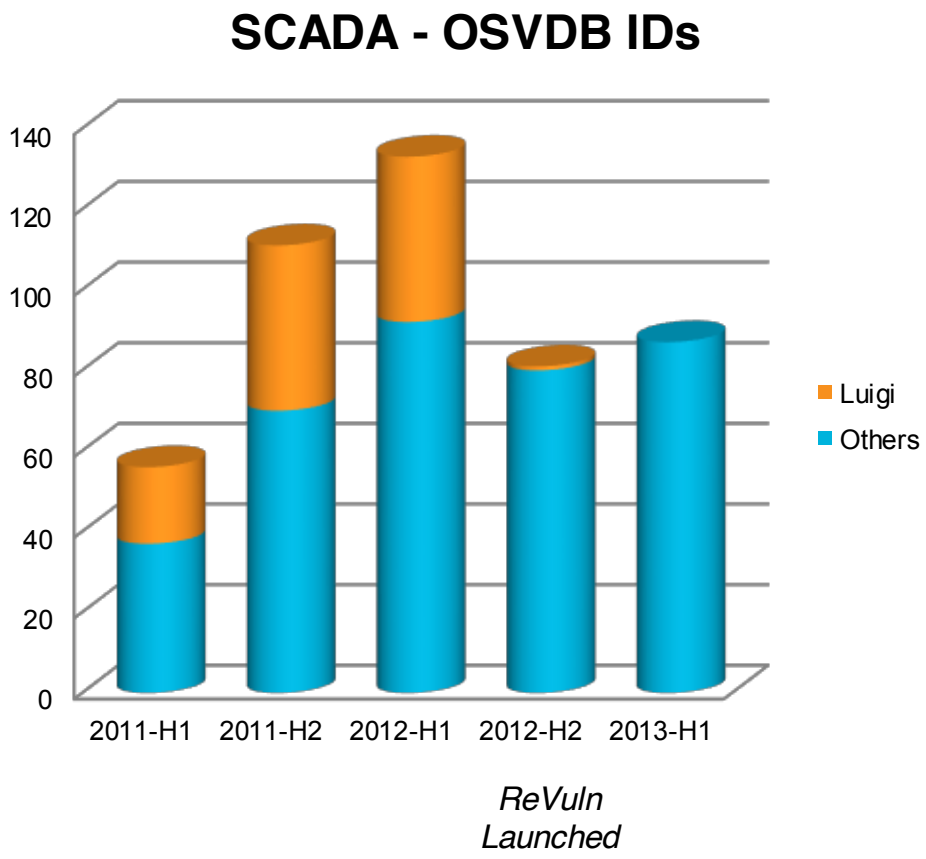
Number of vulns skyrocketed recently

Maybe because of who was looking at it?



MITRE

The Luigi Lossage: Selection & Publication Bias

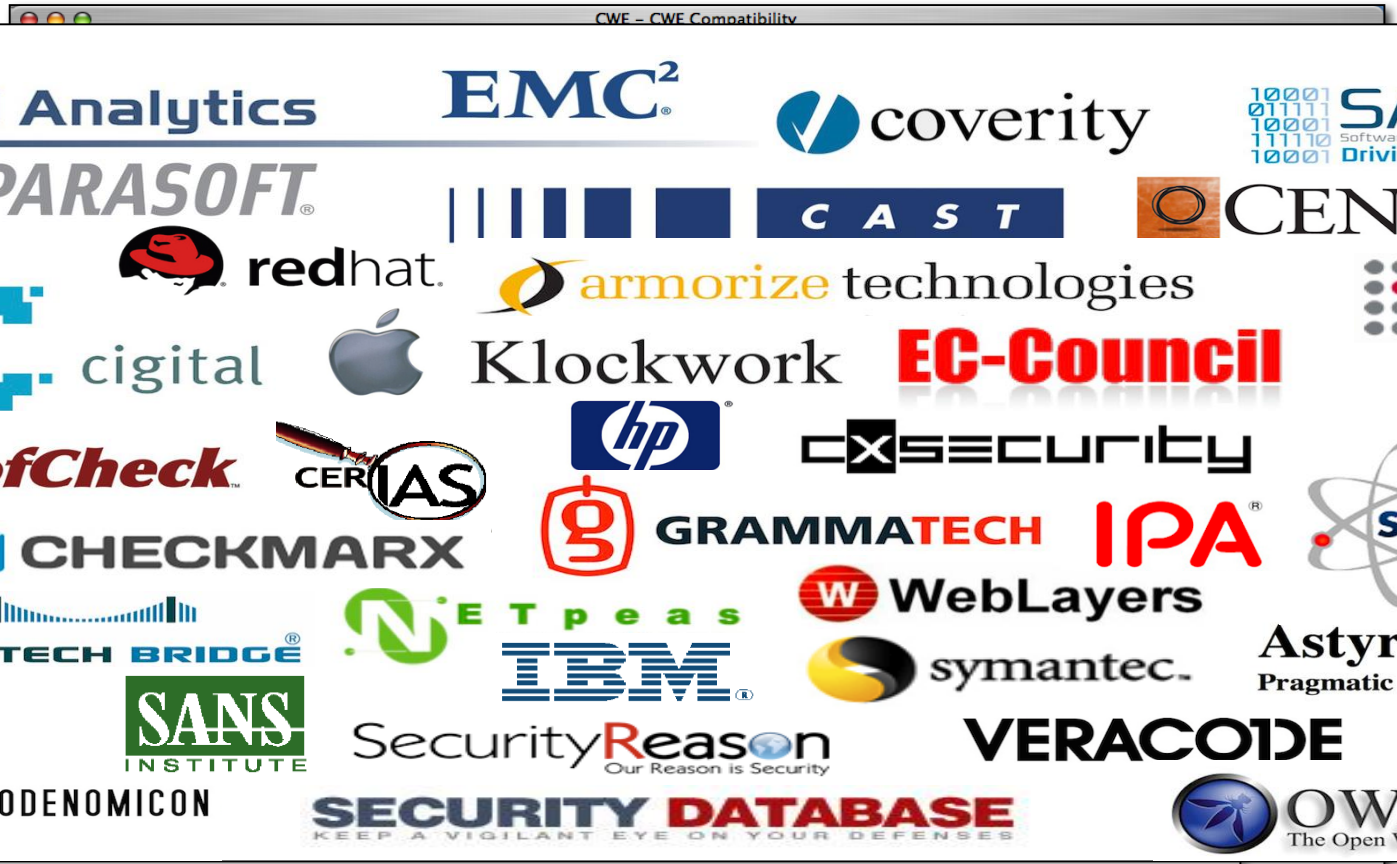


* 2011 Luigi stats may be higher than shown.

MITRE

CWE Compatibility & Effectiveness Program

(launched Feb 2007)



Organizations Participating

cwe.mitre.org/compatible/

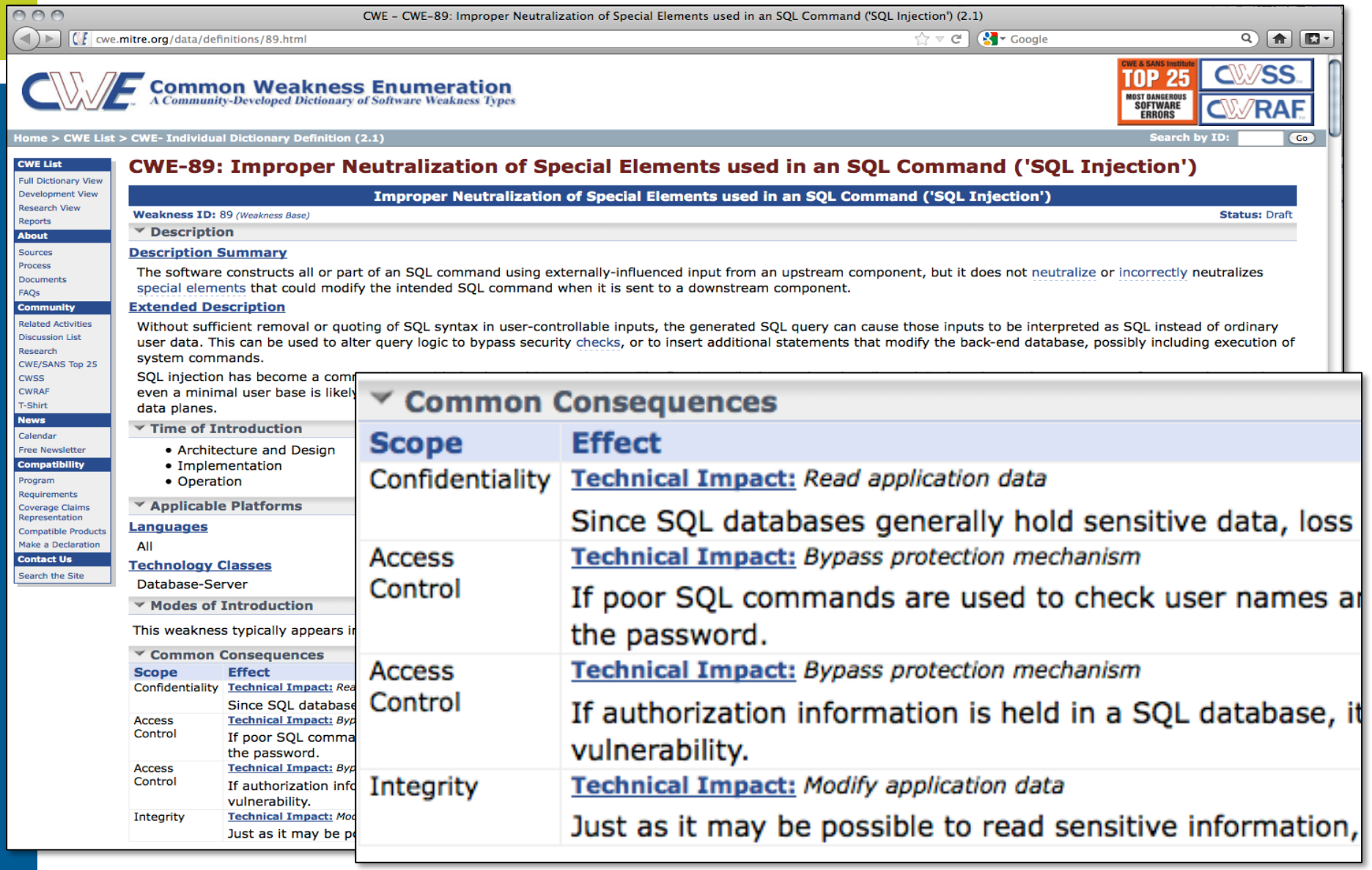
All organizations participating in the CWE Compatibility and Effectiveness Program are listed below, including those with CWE-Compatible Products and Services and those with Declarations to Be CWE-Compatible.

TOTALS
Organizations Participating: 43
Products & Services: 75

Products are listed alphabetically by organization name:

December 29, 2006

Technical Impacts – Common Consequences



MITRE