Standards in reporting Software Flaws: **SCAP**, CVE and CWE (Part 2)

Robin A. Gandhi, Ph.D.

University of Nebraska at Omaha (UNO)
College of Information Science and Technology (IS&T)
School of Interdisciplinary Informatics (Si2)
Nebraska University Center on Information Assurance (NUCIA)

Who am I ?

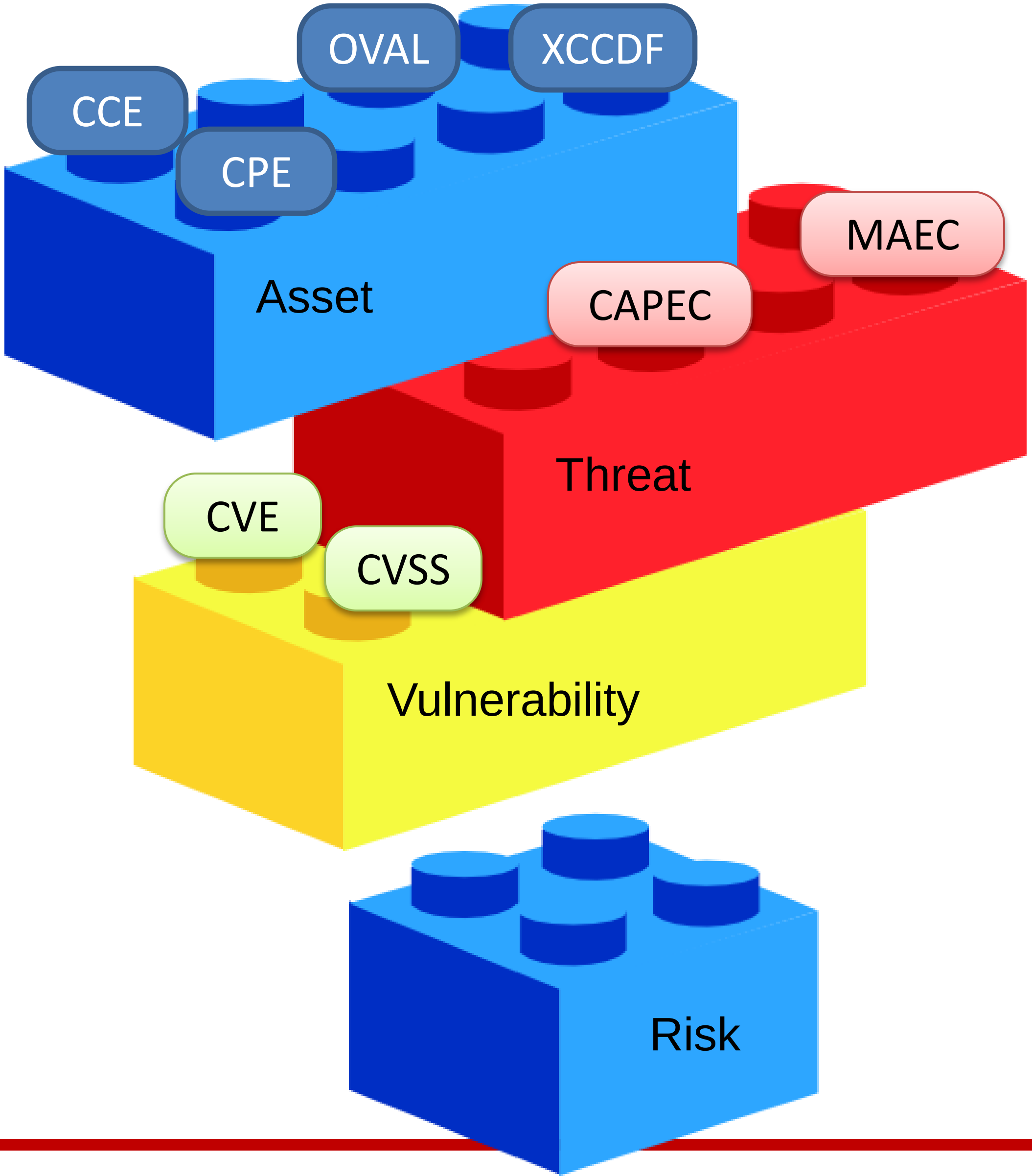
- **Job**
  - Assistant Professor of Information Assurance at IS&T since Fall 2008
- **Research highlights**
  - Regulatory Requirements driven Risk Assessment
    - Using the semantic web to bridge the gap from high-level regulations to low-level technical evidence (Domain: SCADA)
  - Software Assurance in the Development Lifecycle
    - Building semantic templates for the most egregious software flaws
  - Cyber attack modeling and forecasting (CyCast)
    - Exploring disturbances in the human network to predict cyber attacks
- **Teaching**
  - Software Assurance (seniors/grad) New !
  - Foundations of Information Assurance (seniors/grad)
  - Introduction to Information Assurance (Freshmen) New !
  - Introduction to Computer Science II (Freshmen/Sophomore)

A two part talk (Recap Part 1)

- SCAP
  - What is it?
  - What does it do?
  - What will it take to realize its potential?
  - What do I need to do to start preparing for it?

- How can we better understand vulnerabilities
  - Research on semantic templates built from CWE and CVE enumerations

SCAP Philosophy (Recap Part 1)



Different Roles and Responsibilities

- **Information Assurance** professionals tend to focus on the protection of systems that they may NOT have built
  - Extrinsic and deployed view of the system
  - SCAP is geared towards improving the efforts of IA professionals (Vulnerability Assessment/ Hardening)
- **Software Assurance** professionals tend to focus on the development of software systems with security BUILT-IN
  - Intrinsic and functional view of the system
  - Weakness, attack and secure coding enumerations are geared towards improving the efforts of developers

Why Jonny Can't write secure code?

- *Johnny, avoid these weaknesses…. Period!*
  - Common Weaknesses Enumeration (CWE)
- *Johnny…learn from your mistakes*
  - Common Vulnerabilities and Exposures (CVE)
- *Johnny…these are the ways of the bad guys*
  - Common Attack Patterns Enumeration and Classification (CAPEC)
- *Johnny…these are ways to develop secure code*
  - CERT secure coding guidelines

Poor Johnny !

42976
CVE Vulnerabilities

CWE v.19
668 Weaknesses
1043 Pages

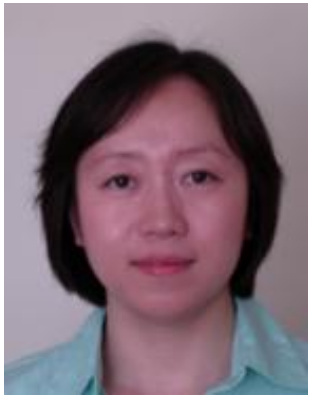
Countless Do's
and Don'ts

CAPEC
311 Attack
Patterns

Using Semantic Templates to Study Vulnerabilities Recorded in Large Software Repositories

Me

Harvey Siy



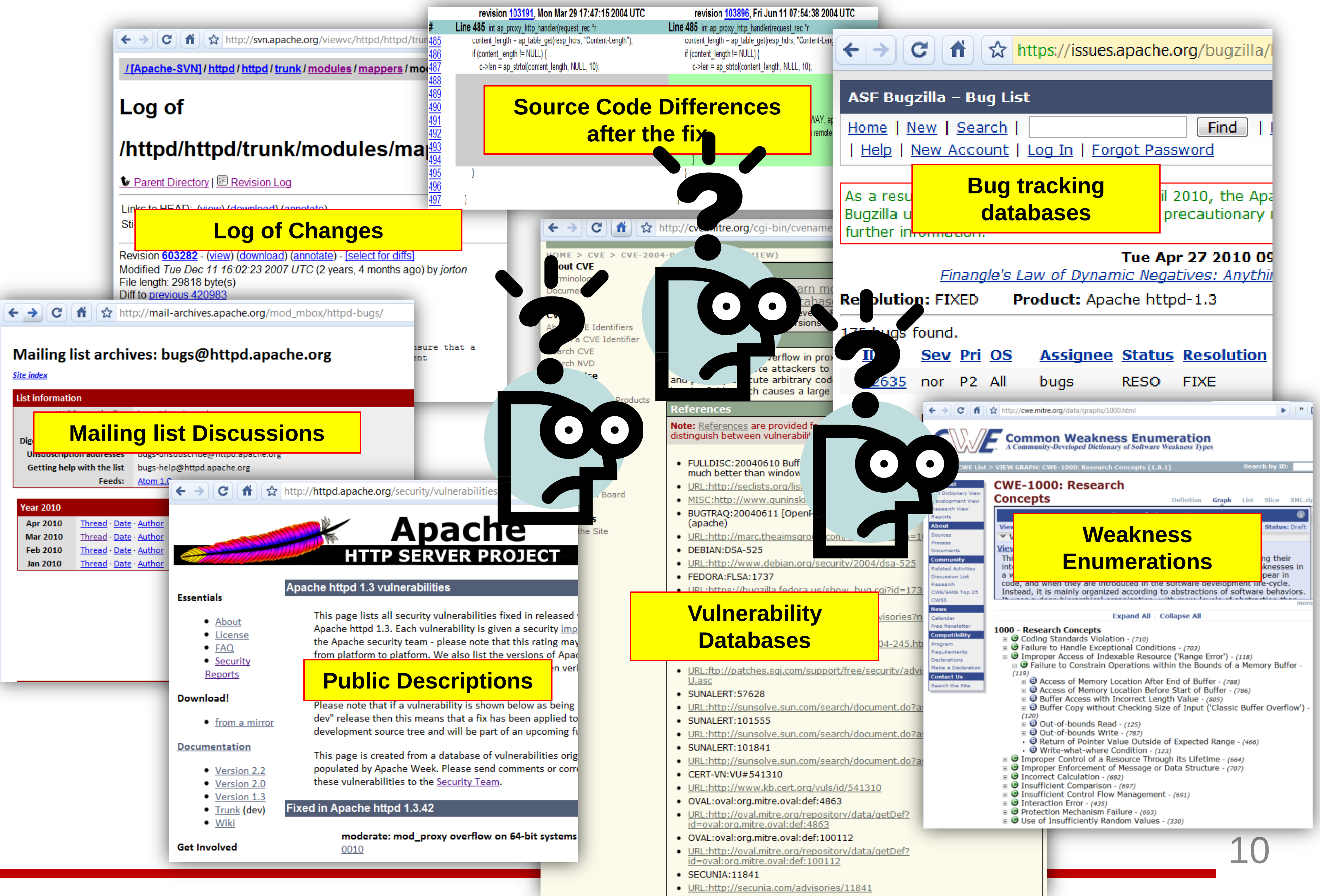
Yan Wu

Outline

- Information overload in the study of vulnerabilities
  - Large software repositories
  - Vulnerability databases
  - Weakness enumerations
- Our research efforts:
  - Building **semantic templates** to understand and categorize the information related to a vulnerability
- Ongoing progress
- Future work

UNIVERSITY OF Nebraska Omaha

The Paradox we face !



Source Code Differences after the fix

Log of Changes

Bug tracking databases

Mailing list Discussions

Public Descriptions

Vulnerability Databases

Weakness Enumerations

http://svn.apache.org/viewvc/httpd/httpd/trun

/ [Apache-SVN] / httpd / httpd / trunk / modules / mappers / mo

Log of

/httpd/httpd/trunk/modules/ma

Parent Directory | Revision Log

Links to HEAD: (view) (download) (annotate)

Revision 603282 - (view) (download) (annotate) - [select for diffs]
Modified Tue Dec 11 16:02:23 2007 UTC (2 years, 4 months ago) by jorton
File length: 29818 byte(s)
Diff to previous 420983

revision 103191, Mon Mar 29 17:47:15 2004 UTC | revision 103896, Fri Jun 11 07:54:38 2004 UTC

```
Line 485 int ap_proxy_http_handler(request_rec *r       Line 485 int ap_proxy_http_handler(request_rec *r
485     content_length = ap_table_get(resp_hdrs, "Content-Length");    content_length = ap_table_get(resp_hdrs, "Content-Leng
486     if (content_length != NULL) {                  if (content_length != NULL) {
487         c->len = ap_strtol(content_length, NULL, 10);   c->len = ap_strtol(content_length, NULL, 10);
488
489
490
491                                                         WAY, ap
492                                                         remote
493
494
495     }
496
497 }
```

https://issues.apache.org/bugzilla/

ASF Bugzilla – Bug List

Home | New | Search | Find | Help | New Account | Log In | Forgot Password

As a resu … il 2010, the Apa Bugzilla u … precautionary further information.

Tue Apr 27 2010 09

Finangle's Law of Dynamic Negatives: Anythi

Resolution: FIXED Product: Apache httpd-1.3

175 bugs found.

ID	Sev	Pri	OS	Assignee	Status	Resolution
2635	nor	P2	All	bugs	RESO	FIXE

http://cve.mitre.org/cgi-bin/cvename

HOME > CVE > CVE-2004-0 … VIEW)

About CVE, Terminology, Documents, CVE Identifiers, a CVE Identifier, Search CVE, Search NVD, Products, Board, the Site

… rflow in prox … te attackers to … cute arbitrary cod … h causes a large

References

Note: References are provided f… distinguish between vulnerabili…

- FULLDISC:20040610 Buff… much better than window…
- URL:http://seclists.org/lis…
- MISC:http://www.guninski…
- BUGTRAQ:20040611 [OpenP… (apache)
- URL:http://marc.theaimsgro…
- DEBIAN:DSA-525
- URL:http://www.debian.org/security/2004/dsa-525
- FEDORA:FLSA:1737
- URL:https://bugzilla.fedora.us/show_bug.cgi?id=173
- URL:ftp://patches.sgi.com/support/free/security/advi… U.asc
- SUNALERT:57628
- URL:http://sunsolve.sun.com/search/document.do?a
- SUNALERT:101555
- URL:http://sunsolve.sun.com/search/document.do?a
- SUNALERT:101841
- URL:http://sunsolve.sun.com/search/document.do?a
- CERT-VN:VU#541310
- URL:http://www.kb.cert.org/vuls/id/541310
- OVAL:oval:org.mitre.oval:def:4863
- URL:http://oval.mitre.org/repository/data/getDef?id=oval:org.mitre.oval:def:4863
- OVAL:oval:org.mitre.oval:def:100112
- URL:http://oval.mitre.org/repository/data/getDef?id=oval:org.mitre.oval:def:100112
- SECUNIA:11841
- URL:http://secunia.com/advisories/11841

http://mail-archives.apache.org/mod_mbox/httpd-bugs/

Mailing list archives: bugs@httpd.apache.org

Site index

List information

Unsubscription addresses	bugs-unsubscribe@httpd.apache.org
Getting help with the list	bugs-help@httpd.apache.org
Feeds:	Atom 1.0

Year 2010	
Apr 2010	Thread · Date · Author
Mar 2010	Thread · Date · Author
Feb 2010	Thread · Date · Author
Jan 2010	Thread · Date · Author

http://httpd.apache.org/security/vulnerabilities

Apache HTTP SERVER PROJECT

Essentials

- About
- License
- FAQ
- Security Reports

Download!

- from a mirror

Documentation

- Version 2.2
- Version 2.0
- Version 1.3
- Trunk (dev)
- Wiki

Get Involved

Apache httpd 1.3 vulnerabilities

This page lists all security vulnerabilities fixed in released Apache httpd 1.3. Each vulnerability is given a security imp… the Apache security team - please note that this rating may… from platform to platform. We also list the versions of Apac… n veri…

Please note that if a vulnerability is shown below as being … dev" release then this means that a fix has been applied to… development source tree and will be part of an upcoming fu…

This page is created from a database of vulnerabilities orig populated by Apache Week. Please send comments or corr these vulnerabilities to the Security Team.

Fixed in Apache httpd 1.3.42

moderate: mod_proxy overflow on 64-bit systems
0010

http://cwe.mitre.org/data/graphs/1000.html

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

CWE List > VIEW GRAPH: CWE-1000: Research Concepts (1.8.1)

CWE-1000: Research Concepts

Status: Draft

… code, and when they are introduced in the software development life-cycle. Instead, it is mainly organized according to abstractions of software behaviors.

Expand All | Collapse All

1000 - Research Concepts
- Coding Standards Violation - (710)
- Failure to Handle Exceptional Conditions - (703)
- Improper Access of Indexable Resource ('Range Error') - (118)
  - Failure to Constrain Operations within the Bounds of a Memory Buffer - (119)
    - Access of Memory Location After End of Buffer - (788)
    - Access of Memory Location Before Start of Buffer - (786)
    - Buffer Access with Incorrect Length Value - (805)
    - Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') - (120)
    - Out-of-bounds Read - (125)
    - Out-of-bounds Write - (787)
    - Return of Pointer Value Outside of Expected Range - (466)
    - Write-what-where Condition - (123)
- Improper Control of a Resource Through its Lifetime - (664)
- Improper Enforcement of Message or Data Structure - (707)
- Incorrect Calculation - (682)
- Insufficient Comparison - (697)
- Insufficient Control Flow Management - (691)
- Interaction Error - (435)
- Protection Mechanism Failure - (693)
- Use of Insufficiently Random Values - (330)

Large Software Repositories

- Source code version control systems (CVS, SVN)
  - Support distributed development
  - Versioning, Merging and Backup functions
  - Huge!
- Log of changes
  - Brief descriptions of the change performed
  - who, when, what, why
- IDEs, Bug tracker databases (reporters, resolvers, discussions), Public websites
- Mailing list threads related to the changes
  - Stakeholders: Developers, Organizations

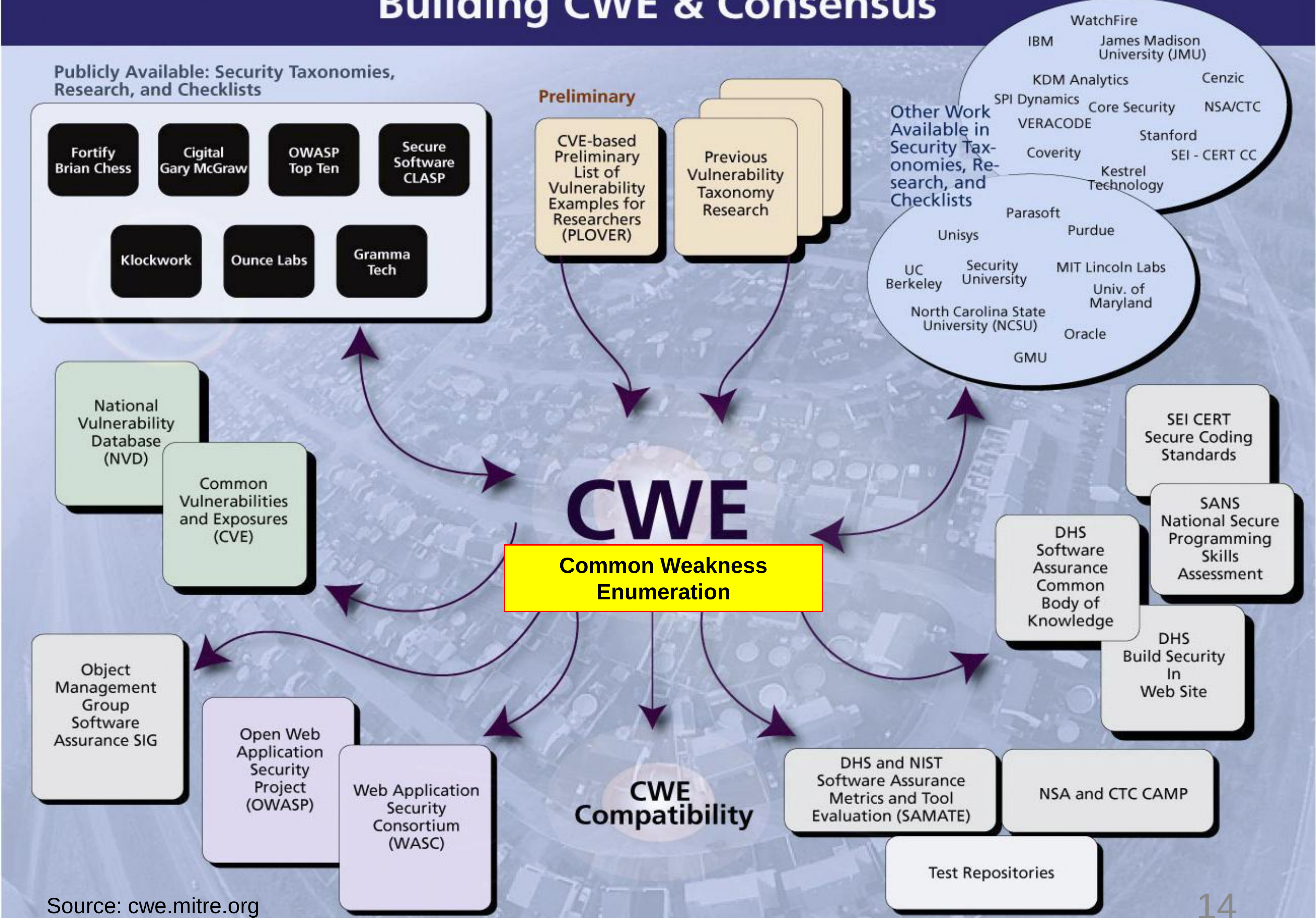
Vulnerability Databases

- Several databases available
  - IBM X-force
  - CERIAS
  - CERT
  - DARPA CIDF
  - BindView Hacker Shield
  - Many others…
- Common Vulnerability Enumeration (CVE)
  - 42976 Vulnerabilities as of 2010-07-21 09:22 CST

Learning from our mistakes

- The Landwehr Software Flaw Taxonomy (1993)
  - Genesis (How), Time of introduction (When), Location (Where)
- Several recent efforts have followed
  - Seven Pernicious Kingdoms, PLOVER, 19 Deadly Sins, OWASP top ten…
- The Common Weaknesses Enumeration (CWE) has tried to assimilate these efforts and bring consensus (http://cwe.mitre.org)

Building CWE & Consensus



Publicly Available: Security Taxonomies, Research, and Checklists

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

Preliminary

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

Other Work Available in Security Taxonomies, Research, and Checklists

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

CWE

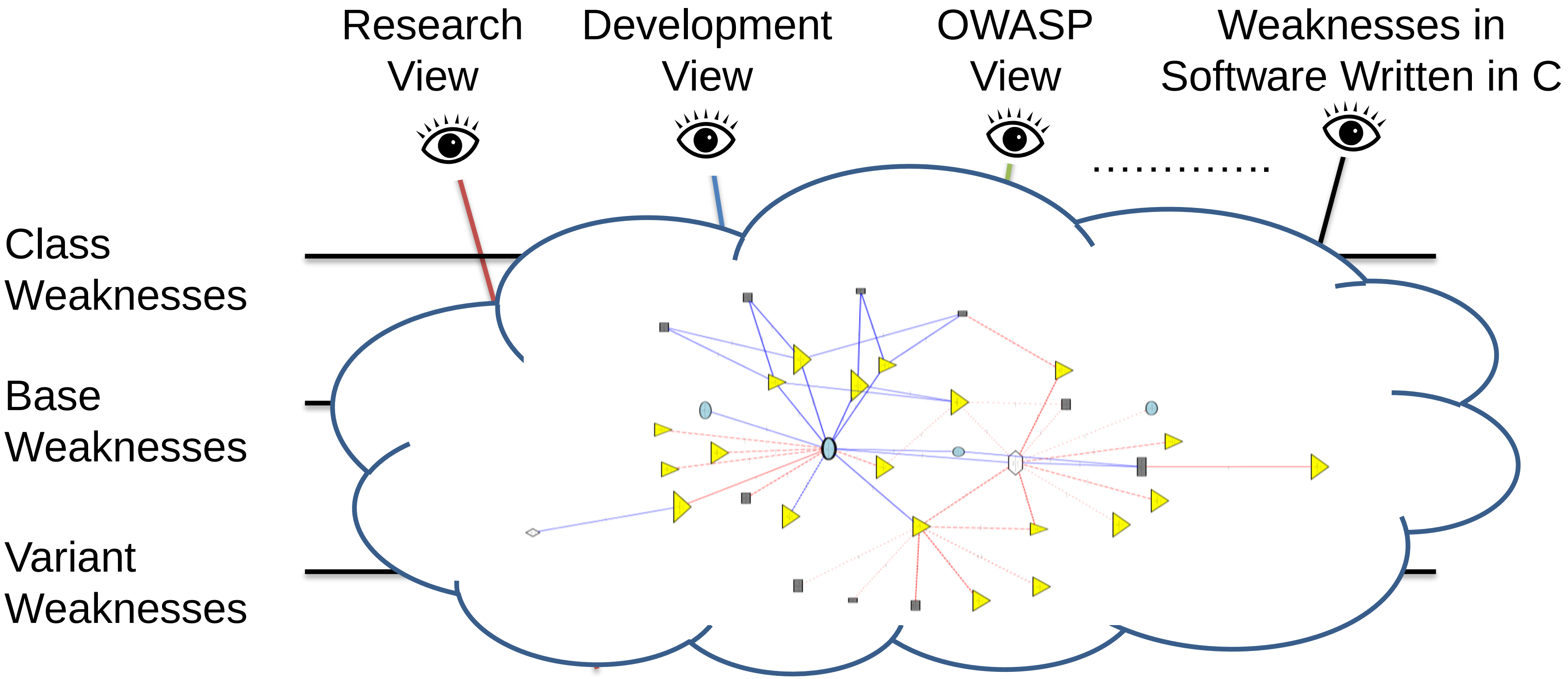
Common Weakness Enumeration

- National Vulnerability Database (NVD)
- Common Vulnerabilities and Exposures (CVE)
- Object Management Group Software Assurance SIG
- Open Web Application Security Project (OWASP)
- Web Application Security Consortium (WASC)
- SEI CERT Secure Coding Standards
- SANS National Secure Programming Skills Assessment
- DHS Software Assurance Common Body of Knowledge
- DHS Build Security In Web Site
- DHS and NIST Software Assurance Metrics and Tool Evaluation (SAMATE)
- NSA and CTC CAMP
- Test Repositories

CWE Compatibility

Source: cwe.mitre.org

CWE Organization (rough)



Weakness Enumerations

- Common Weaknesses Enumeration (CWE)
  - (**measurement**) Unified, measurable set of software weaknesses
  - (**communication**) Effective sharing, description, selection, and use of software security tools and services
  - (**management, prioritization**) Better understanding and management of software weaknesses related to design and code

Problems, Problems, Problems…

- Most vulnerability related artifacts are in unstructured text
  - Makes aggregation of these artifacts harder
- No shortage of weakness enumerations and categorization
  - Adoption in projects is slow
    - many choices could be a factor
- Growing software complexity
  - Little or no effort to improve the mental model of the software developer to sense the possibility of a vulnerability

Reducing the Cognitive Overload

- Devil is in the Details
  - The details about vulnerabilities are enormous during the coding phases
- Simple guides can be more effective than a long checklist
  - The 3 golden questions to ask about each bug (1989)
    - Is this mistake somewhere else also?
    - What next bug is hidden behind this one?
    - What should I do to prevent bugs like this?

Reducing the Cognitive Overload

- Questions about security weaknesses
  - What are the **Software flaws** (commission, omission, operational) that lead to the weakness?
  - What are the defining characteristics of the **Weakness**?
  - What are the ***Resources***/***Location*** where the weakness is typically manifest?
  - What are the **Consequences** that the weakness precedes?

Tangling of information in the CWE

- **CWE-119: Failure to Constrain Operations within the Bounds of a *Memory Buffer***
  - The software performs operations on a ***memory buffer***, but it can read from or write to a memory location that is outside of the intended boundary of the ***buffer***.
  - Certain languages allow direct addressing of memory locations and do not automatically ensure that these locations are valid for the memory buffer that is being referenced. This can cause read or write operations to be performed on memory locations that may be associated with other variables, data structures, or internal program data. As a result, an attacker may be able to execute arbitrary code, alter the intended control flow, read sensitive information, or cause the system to crash.

Tangling of information in the CWE

- **CWE-119: Failure to Constrain Operations within the Bounds of a *Memory Buffer***
  - The software performs operations on a ***memory buffer***, but it can read from or write to a memory location that is outside of the intended boundary of the ***buffer***.
  - Certain languages allow direct addressing of memory locations and do not automatically ensure that these locations are valid for the memory buffer that is being referenced. This can cause read or write operations to be performed on memory locations that may be associated with other variables, data structures, or internal program data. As a result, an attacker may be able to execute arbitrary code, alter the intended control flow, read sensitive information, or cause the system to crash.

LEGEND

Software Fault

Weakness

Resource/Location

Consequence

Tangling of information in the CWE

- CWE-120: ***Buffer*** **Copy without Checking Size of Input** ('Classic **Buffer Overflow**')
  - The program copies an input ***buffer*** to an output ***buffer*** without verifying that the size of the input ***buffer*** is less than the size of the output ***buffer***, leading to a **buffer overflow.**
  - A **buffer overflow** condition exists when a program attempts to put more data in a ***buffer*** than it can hold, or when a program attempts to put data in a ***memory area*** outside of the boundaries of a ***buffer***.
  - **Buffer overflows** often can be used to **execute arbitrary code**…
  - **Buffer overflows** generally **lead to crashes**

LEGEND

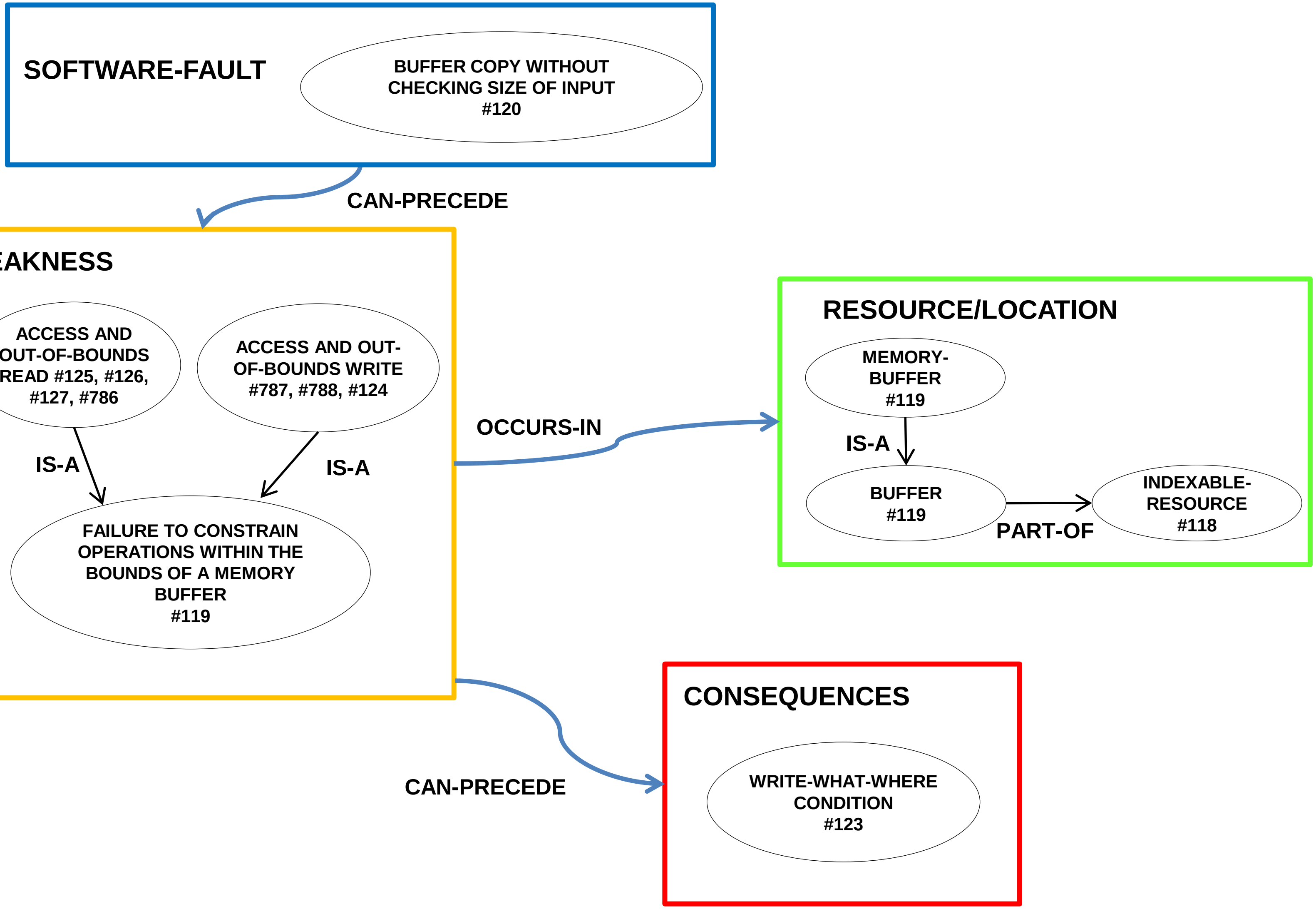
Software Fault

Weakness

Resource/Location

Consequence

UnTangling


SOFTWARE-FAULT
BUFFER COPY WITHOUT CHECKING SIZE OF INPUT #120
CAN-PRECEDE
WEAKNESS
ACCESS AND OUT-OF-BOUNDS READ #125, #126, #127, #786
ACCESS AND OUT-OF-BOUNDS WRITE #787, #788, #124
IS-A
IS-A
FAILURE TO CONSTRAIN OPERATIONS WITHIN THE BOUNDS OF A MEMORY BUFFER #119
OCCURS-IN
RESOURCE/LOCATION
MEMORY-BUFFER #119
IS-A
BUFFER #119
PART-OF
INDEXABLE-RESOURCE #118
CAN-PRECEDE
CONSEQUENCES
WRITE-WHAT-WHERE CONDITION #123

Building a Semantic Template

- For each weakness type create a semantic template

CE Concept Extraction

- Exploration of the CWE structure to extract entries relevant to a weakness

TS Template structuring

- Software Fault, Weakness, Resources, Consequences

TR Template refinement

- Aggregation of Vulnerability Artifacts
- Annotation using Semantic template concepts

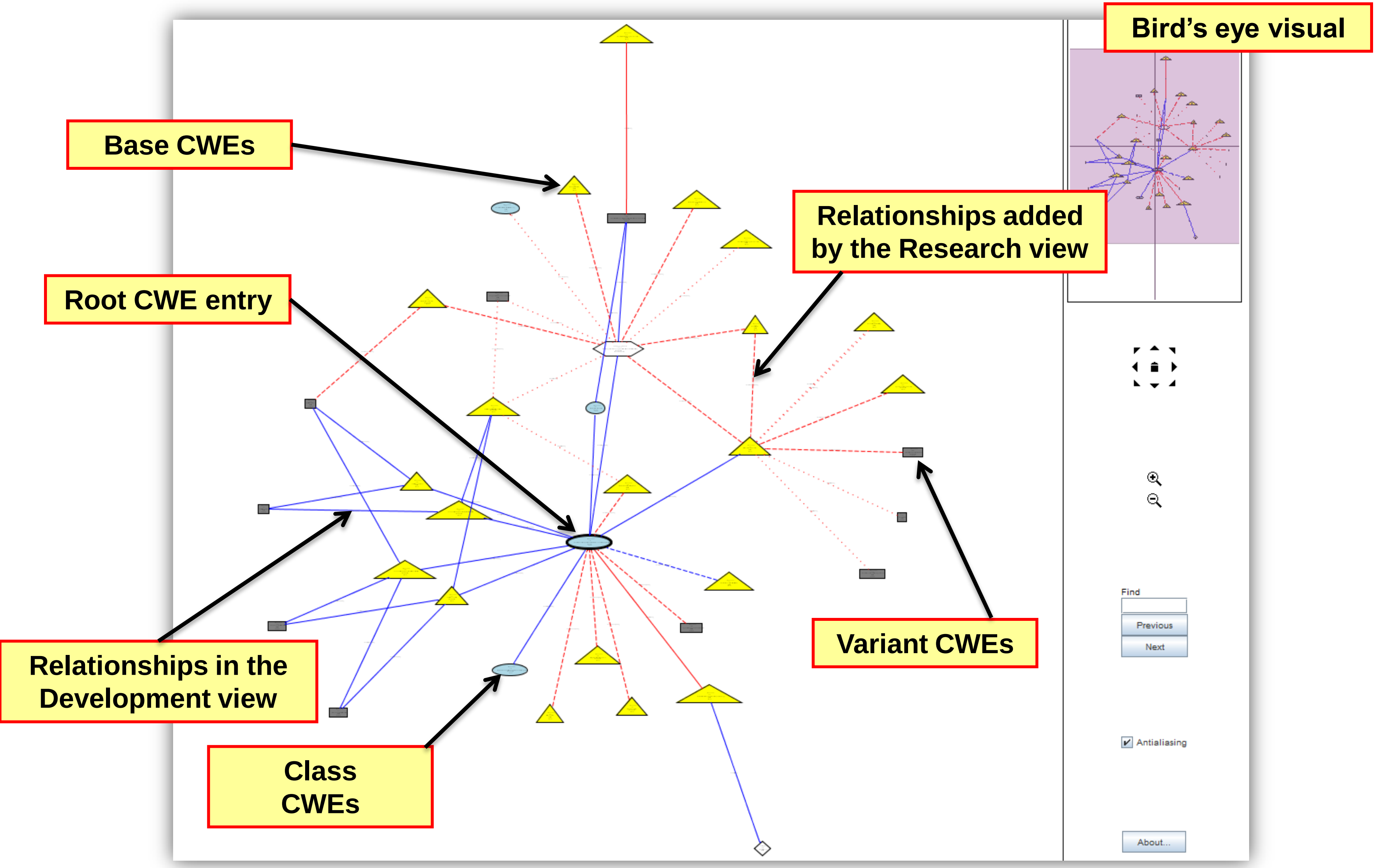
Concept Extraction

- CWE 1.6
- Development view
  - – Suited for stakeholders in the SDLC
- Research view
  - – Suited for research using the cwe; deep hierarchical structure
- Select a “Root entry”
  - – CWE that provides the most abstract description of a weakness, that would be CWE 119 for BO

Concept Extraction

- Strategies:
  1. Navigate hierarchical relationships of the root entry
  2. Navigate non-taxonomical relationships such as “Can Precede”, “Can Follow”, “Peer-of”
  3. Visualization of the CWE XML specification
     - A graph is generated using graphviz
  4. Keyword search on the CWE hyperlinked document
     - Followed by exploration of parent, sibling and child categories of the discovered CWE, for relevance to the root entry

Visualization



Bird's eye visual

Base CWEs

Relationships added by the Research view

Root CWE entry

Variant CWEs

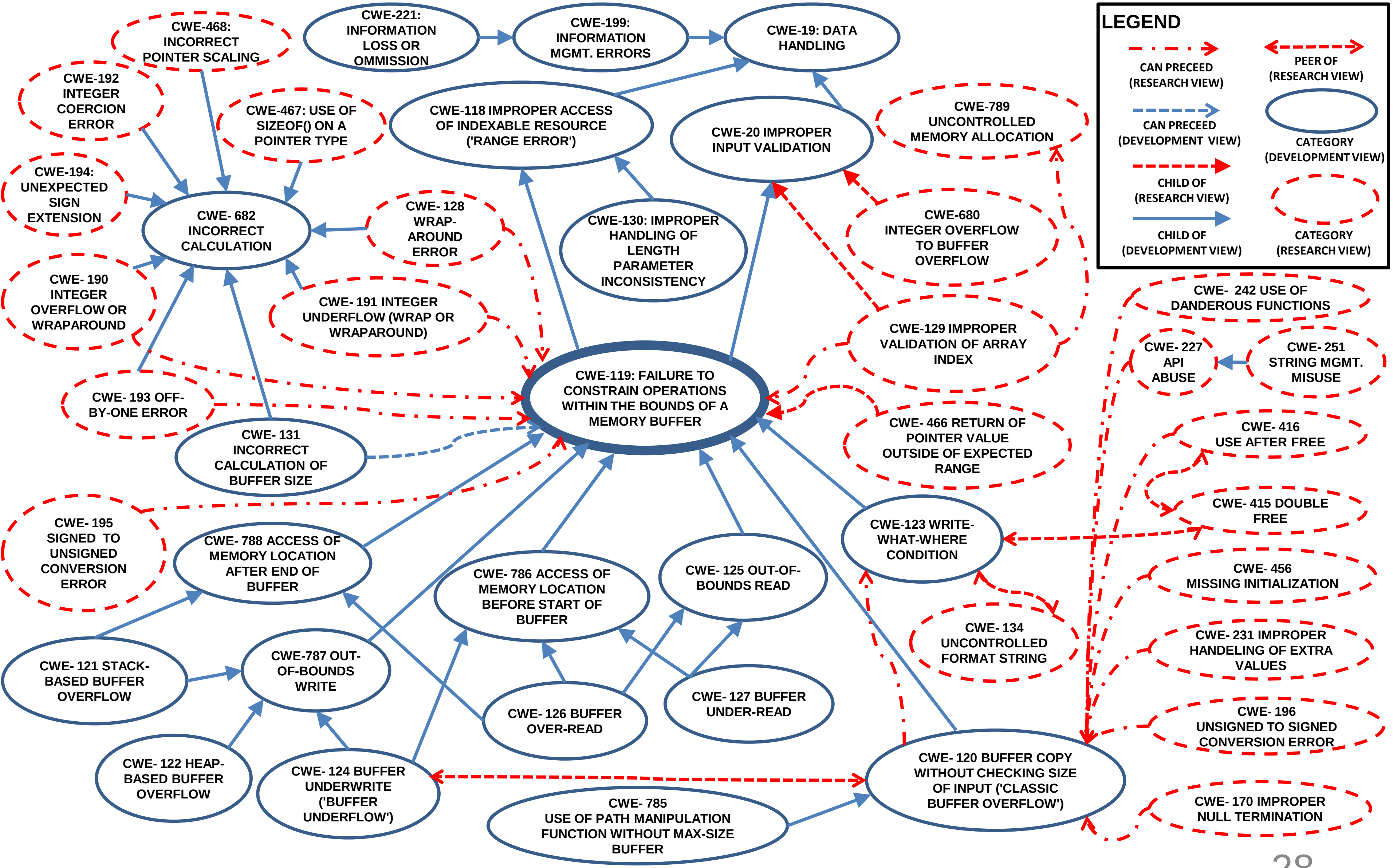
Relationships in the Development view

Class CWEs

☑ Antialiasing

http://www.cs.unomaha.edu/~hsiy/research/zgrview/boverflowCWEs.html

Concept Extraction


LEGEND
CAN PRECEED (RESEARCH VIEW)
PEER OF (RESEARCH VIEW)
CAN PRECEED (DEVELOPMENT VIEW)
CATEGORY (DEVELOPMENT VIEW)
CHILD OF (RESEARCH VIEW)
CHILD OF (DEVELOPMENT VIEW)
CATEGORY (RESEARCH VIEW)
CWE-468: INCORRECT POINTER SCALING
CWE-221: INFORMATION LOSS OR OMMISSION
CWE-199: INFORMATION MGMT. ERRORS
CWE-19: DATA HANDLING
CWE-192 INTEGER COERCION ERROR
CWE-467: USE OF SIZEOF() ON A POINTER TYPE
CWE-118 IMPROPER ACCESS OF INDEXABLE RESOURCE ('RANGE ERROR')
CWE-20 IMPROPER INPUT VALIDATION
CWE-789 UNCONTROLLED MEMORY ALLOCATION
CWE-194: UNEXPECTED SIGN EXTENSION
CWE- 682 INCORRECT CALCULATION
CWE- 128 WRAP-AROUND ERROR
CWE-130: IMPROPER HANDLING OF LENGTH PARAMETER INCONSISTENCY
CWE-680 INTEGER OVERFLOW TO BUFFER OVERFLOW
CWE- 190 INTEGER OVERFLOW OR WRAPAROUND
CWE- 191 INTEGER UNDERFLOW (WRAP OR WRAPAROUND)
CWE-129 IMPROPER VALIDATION OF ARRAY INDEX
CWE- 242 USE OF DANDEROUS FUNCTIONS
CWE- 227 API ABUSE
CWE- 251 STRING MGMT. MISUSE
CWE- 193 OFF-BY-ONE ERROR
CWE-119: FAILURE TO CONSTRAIN OPERATIONS WITHIN THE BOUNDS OF A MEMORY BUFFER
CWE- 131 INCORRECT CALCULATION OF BUFFER SIZE
CWE- 466 RETURN OF POINTER VALUE OUTSIDE OF EXPECTED RANGE
CWE- 416 USE AFTER FREE
CWE- 195 SIGNED TO UNSIGNED CONVERSION ERROR
CWE- 788 ACCESS OF MEMORY LOCATION AFTER END OF BUFFER
CWE- 415 DOUBLE FREE
CWE-123 WRITE-WHAT-WHERE CONDITION
CWE- 786 ACCESS OF MEMORY LOCATION BEFORE START OF BUFFER
CWE- 125 OUT-OF-BOUNDS READ
CWE- 456 MISSING INITIALIZATION
CWE- 134 UNCONTROLLED FORMAT STRING
CWE- 121 STACK-BASED BUFFER OVERFLOW
CWE-787 OUT-OF-BOUNDS WRITE
CWE- 231 IMPROPER HANDELING OF EXTRA VALUES
CWE- 126 BUFFER OVER-READ
CWE- 127 BUFFER UNDER-READ
CWE- 196 UNSIGNED TO SIGNED CONVERSION ERROR
CWE- 122 HEAP-BASED BUFFER OVERFLOW
CWE- 124 BUFFER UNDERWRITE ('BUFFER UNDERFLOW')
CWE- 120 BUFFER COPY WITHOUT CHECKING SIZE OF INPUT ('CLASSIC BUFFER OVERFLOW')
CWE- 785 USE OF PATH MANIPULATION FUNCTION WITHOUT MAX-SIZE BUFFER
CWE- 170 IMPROPER NULL TERMINATION

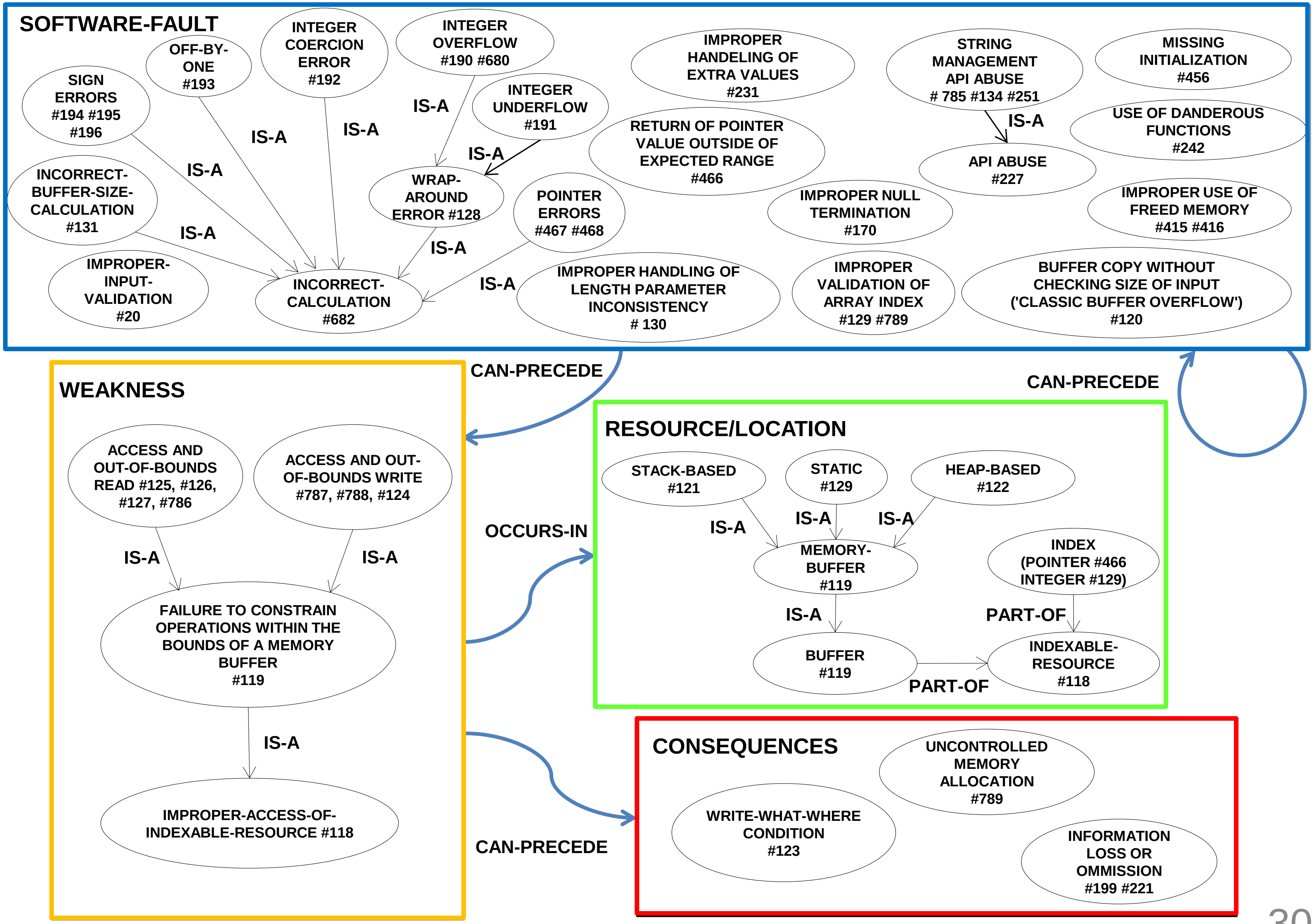
Template Structuring

- Each CWE identified in the previous step is analyzed for concepts along the conceptual unit of the semantic template
- Relationships among the CWE entries are then used to structure the identified concepts into a coherent semantic template

- CWE-120: ***Buffer*** **Copy without Checking Size of Input** ('Classic **Buffer Overflow**')
  - The program copies an input ***buffer*** to an output ***buffer*** without verifying that the size of the input ***buffer*** is less than the size of the output ***buffer***, leading to a **buffer overflow**.
  - A **buffer overflow** condition exists when a program attempts to put more data in a ***buffer*** than it can hold, or when a program attempts to put data in a ***memory area*** outside of the boundaries of a ***buffer***.
  - **Buffer overflows** often can be used to **execute arbitrary code**…
  - **Buffer overflows** generally **lead to crashes**

Buffer Overflow

CE TS TR



SOFTWARE-FAULT

SIGN ERRORS #194 #195 #196

OFF-BY-ONE #193

INTEGER COERCION ERROR #192

INTEGER OVERFLOW #190 #680

INTEGER UNDERFLOW #191

WRAP-AROUND ERROR #128

POINTER ERRORS #467 #468

INCORRECT-BUFFER-SIZE-CALCULATION #131

IMPROPER-INPUT-VALIDATION #20

INCORRECT-CALCULATION #682

IMPROPER HANDELING OF EXTRA VALUES #231

RETURN OF POINTER VALUE OUTSIDE OF EXPECTED RANGE #466

IMPROPER HANDLING OF LENGTH PARAMETER INCONSISTENCY # 130

IMPROPER NULL TERMINATION #170

IMPROPER VALIDATION OF ARRAY INDEX #129 #789

STRING MANAGEMENT API ABUSE # 785 #134 #251

API ABUSE #227

MISSING INITIALIZATION #456

USE OF DANDEROUS FUNCTIONS #242

IMPROPER USE OF FREED MEMORY #415 #416

BUFFER COPY WITHOUT CHECKING SIZE OF INPUT ('CLASSIC BUFFER OVERFLOW') #120

WEAKNESS

ACCESS AND OUT-OF-BOUNDS READ #125, #126, #127, #786

ACCESS AND OUT-OF-BOUNDS WRITE #787, #788, #124

FAILURE TO CONSTRAIN OPERATIONS WITHIN THE BOUNDS OF A MEMORY BUFFER #119

IMPROPER-ACCESS-OF-INDEXABLE-RESOURCE #118

RESOURCE/LOCATION

STACK-BASED #121

STATIC #129

HEAP-BASED #122

MEMORY-BUFFER #119

INDEX (POINTER #466 INTEGER #129)

BUFFER #119

INDEXABLE-RESOURCE #118

CONSEQUENCES

UNCONTROLLED MEMORY ALLOCATION #789

WRITE-WHAT-WHERE CONDITION #123

INFORMATION LOSS OR OMMISSION #199 #221

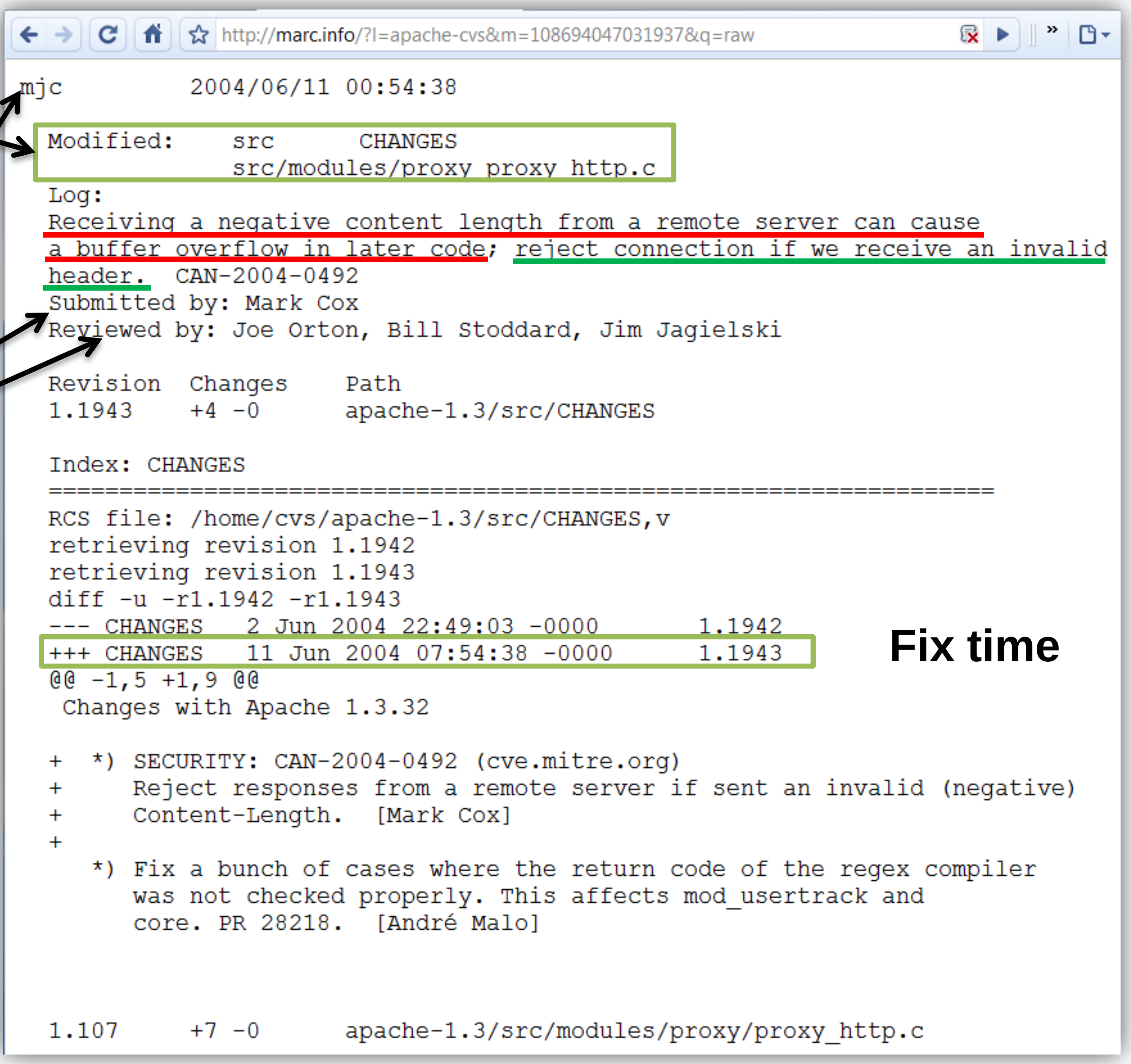
Apache HTTP Server

- Widely used web server
- Open Source project with a large software repository readily available
- Due to the project size and its complexity, various vulnerabilities have occurred and solved during its lifetime

CVE (CAN-2004-0492)

- National Vulnerability Database **(Vulnerability Database)** http://web.nvd.nist.gov/view/vuln/detail?vulnId=CVE-2004-0492
    - **Heap-based buffer overflow** in **proxy_util.c** for **mod_proxy** in **Apache 1.3.25 to 1.3.31** allows remote attackers to cause **a denial of service (process crash)** and possibly **execute arbitrary code via a negative Content-Length HTTP header field**, which causes a large amount of data to be copied.
- Apache Security Reports **(Public website)**
- http://httpd.apache.org/security/vulnerabilities_13.html
    - A **buffer overflow** was found in the **Apache proxy module, mod_proxy**, which can be triggered by receiving an **invalid Content-Length header**. In order to exploit this issue an attacker would need to get an Apache installation that was configured as a proxy to connect to a malicious site. This would cause the Apache child processing the request to crash, although this does not represent a significant Denial of Service attack as requests will continue to be handled by other Apache child processes. This issue may lead to remote arbitrary code execution on some BSD platforms.

Apache Log of Changes(Fix)



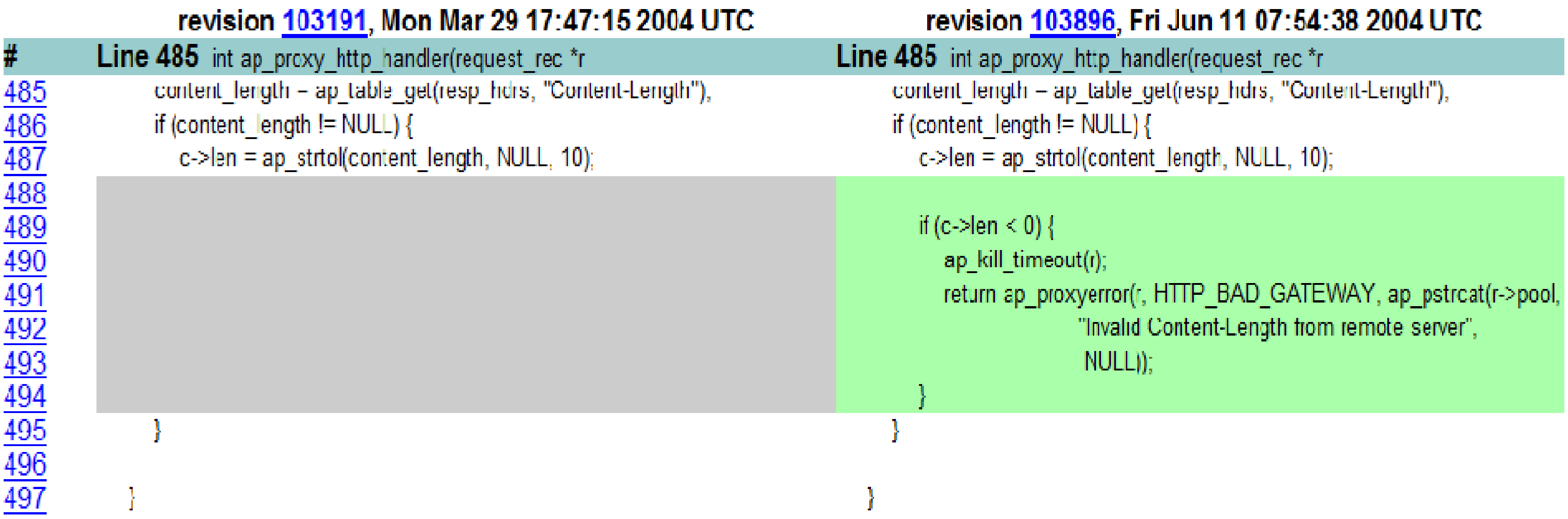
Files Fixed in Project:

Roles

Fix time

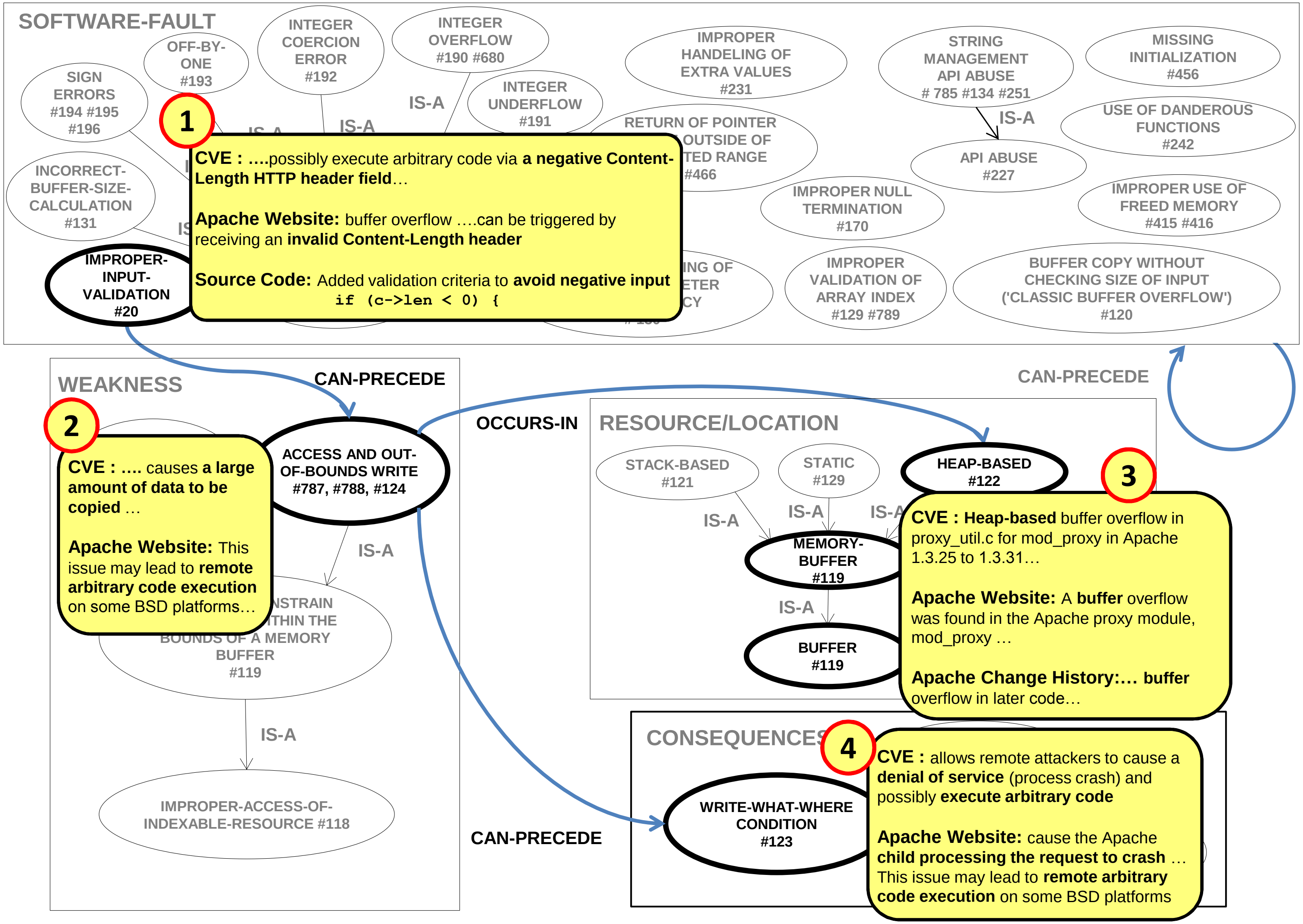
```
mjc         2004/06/11 00:54:38

  Modified:    src       CHANGES
               src/modules/proxy proxy_http.c
  Log:
  Receiving a negative content length from a remote server can cause
  a buffer overflow in later code; reject connection if we receive an invalid
  header.  CAN-2004-0492
  Submitted by: Mark Cox
  Reviewed by: Joe Orton, Bill Stoddard, Jim Jagielski

  Revision  Changes    Path
  1.1943    +4 -0      apache-1.3/src/CHANGES

  Index: CHANGES
  ===================================================================
  RCS file: /home/cvs/apache-1.3/src/CHANGES,v
  retrieving revision 1.1942
  retrieving revision 1.1943
  diff -u -r1.1942 -r1.1943
  --- CHANGES   2 Jun 2004 22:49:03 -0000       1.1942
  +++ CHANGES   11 Jun 2004 07:54:38 -0000      1.1943
  @@ -1,5 +1,9 @@
   Changes with Apache 1.3.32

  +  *) SECURITY: CAN-2004-0492 (cve.mitre.org)
  +     Reject responses from a remote server if sent an invalid (negative)
  +     Content-Length.  [Mark Cox]
  +
     *) Fix a bunch of cases where the return code of the regex compiler
        was not checked properly. This affects mod_usertrack and
        core. PR 28218.  [André Malo]



  1.107     +7 -0      apache-1.3/src/modules/proxy/proxy_http.c
```

Source Code Differences



#	revision 103191, Mon Mar 29 17:47:15 2004 UTC	revision 103896, Fri Jun 11 07:54:38 2004 UTC
	Line 485 int ap_proxy_http_handler(request_rec *r	**Line 485** int ap_proxy_http_handler(request_rec *r
485	content_length = ap_table_get(resp_hdrs, "Content-Length");	content_length = ap_table_get(resp_hdrs, "Content-Length");
486	if (content_length != NULL) {	if (content_length != NULL) {
487	c->len = ap_strtol(content_length, NULL, 10);	c->len = ap_strtol(content_length, NULL, 10);
488		
489		if (c->len < 0) {
490		ap_kill_timeout(r);
491		return ap_proxyerror(r, HTTP_BAD_GATEWAY, ap_pstrcat(r->pool,
492		"Invalid Content-Length from remote server",
493		NULL));
494		}
495	}	}
496		
497	}	}

Study of the Vulnerability


SOFTWARE-FAULT
SIGN ERRORS #194 #195 #196
OFF-BY-ONE #193
INTEGER COERCION ERROR #192
INTEGER OVERFLOW #190 #680
INTEGER UNDERFLOW #191
IMPROPER HANDELING OF EXTRA VALUES #231
RETURN OF POINTER OUTSIDE OF ...TED RANGE #466
STRING MANAGEMENT API ABUSE # 785 #134 #251
API ABUSE #227
MISSING INITIALIZATION #456
USE OF DANDEROUS FUNCTIONS #242
IMPROPER NULL TERMINATION #170
IMPROPER USE OF FREED MEMORY #415 #416
INCORRECT-BUFFER-SIZE-CALCULATION #131
IMPROPER-INPUT-VALIDATION #20
...ING OF ...ETER ...CY
IMPROPER VALIDATION OF ARRAY INDEX #129 #789
BUFFER COPY WITHOUT CHECKING SIZE OF INPUT ('CLASSIC BUFFER OVERFLOW') #120
IS-A
IS-A
IS-A
IS-A
1
WEAKNESS
CAN-PRECEDE
ACCESS AND OUT-OF-BOUNDS WRITE #787, #788, #124
IS-A
...NSTRAIN ...THIN THE BOUNDS OF A MEMORY BUFFER #119
IS-A
IMPROPER-ACCESS-OF-INDEXABLE-RESOURCE #118
2
OCCURS-IN
CAN-PRECEDE
RESOURCE/LOCATION
STACK-BASED #121
STATIC #129
HEAP-BASED #122
IS-A
IS-A
IS-A
MEMORY-BUFFER #119
IS-A
BUFFER #119
3
CONSEQUENCES
4
WRITE-WHAT-WHERE CONDITION #123
CAN-PRECEDE

CVE : ….possibly execute arbitrary code via **a negative Content-Length HTTP header field**…

Apache Website: buffer overflow ….can be triggered by receiving an **invalid Content-Length header**

Source Code: Added validation criteria to **avoid negative input**

```
if (c->len < 0) {
```

CVE : …. causes **a large amount of data to be copied** …

Apache Website: This issue may lead to **remote arbitrary code execution** on some BSD platforms…

CVE : Heap-based buffer overflow in proxy_util.c for mod_proxy in Apache 1.3.25 to 1.3.31…

Apache Website: A **buffer** overflow was found in the Apache proxy module, mod_proxy …

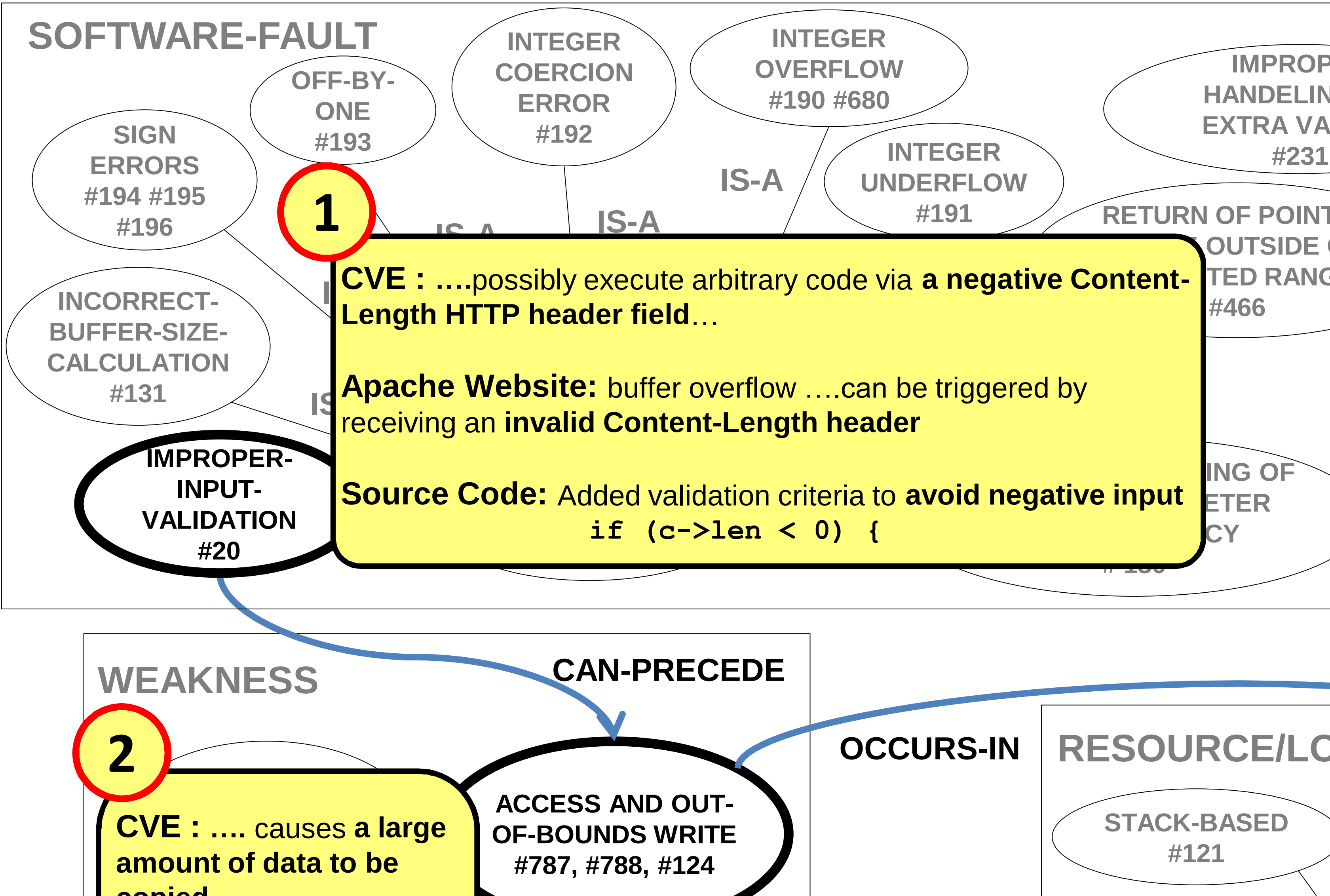
Apache Change History:… buffer overflow in later code…

CVE : allows remote attackers to cause a **denial of service** (process crash) and possibly **execute arbitrary code**

Apache Website: cause the Apache **child processing the request to crash** … This issue may lead to **remote arbitrary code execution** on some BSD platforms

CE TS TR

Study of the Vulnerability


SOFTWARE-FAULT
SIGN ERRORS #194 #195 #196
OFF-BY-ONE #193
INTEGER COERCION ERROR #192
INTEGER OVERFLOW #190 #680
INTEGER UNDERFLOW #191
IMPROP HANDELIN EXTRA VA #231
RETURN OF POINT OUTSIDE TED RANG #466
IS-A
IS-A
IS-A
INCORRECT-BUFFER-SIZE-CALCULATION #131
IMPROPER-INPUT-VALIDATION #20
ING OF ETER CY
1
WEAKNESS
CAN-PRECEDE
2
ACCESS AND OUT-OF-BOUNDS WRITE #787, #788, #124
OCCURS-IN
RESOURCE/LO
STACK-BASED #121

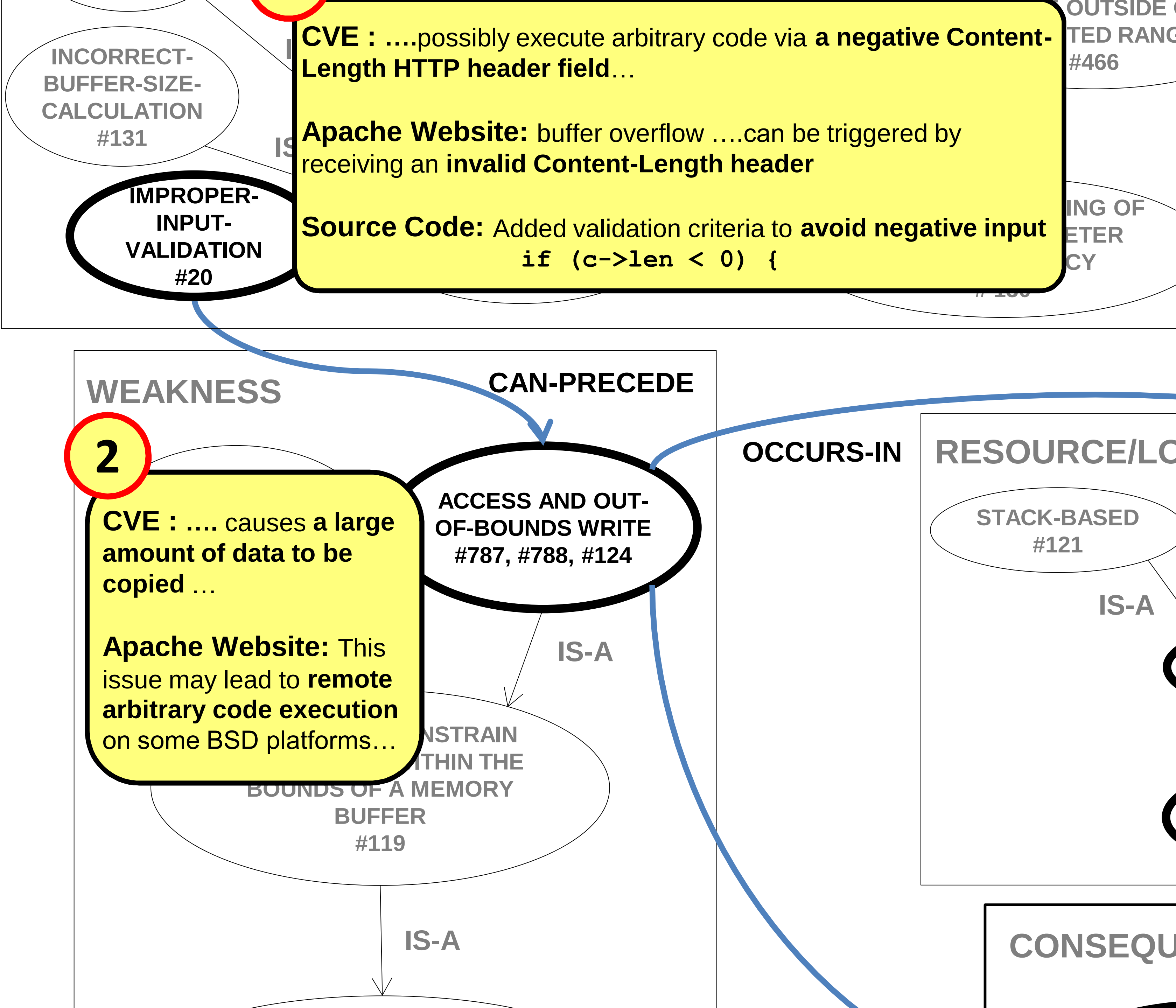
CVE : ….possibly execute arbitrary code via **a negative Content-Length HTTP header field**…

Apache Website: buffer overflow ….can be triggered by receiving an **invalid Content-Length header**

Source Code: Added validation criteria to **avoid negative input**

```
if (c->len < 0) {
```

CVE : …. causes **a large amount of data to be copied**



INCORRECT-BUFFER-SIZE-CALCULATION #131
IS
IMPROPER-INPUT-VALIDATION #20
OUTSIDE
TED RANG
#466
ING OF
ETER
CY
WEAKNESS
CAN-PRECEDE
ACCESS AND OUT-OF-BOUNDS WRITE #787, #788, #124
IS-A
NSTRAIN
ITHIN THE
BOUNDS OF A MEMORY BUFFER #119
IS-A
OCCURS-IN
RESOURCE/LO
STACK-BASED #121
IS-A
CONSEQU
2

CVE : ….possibly execute arbitrary code via **a negative Content-Length HTTP header field**…

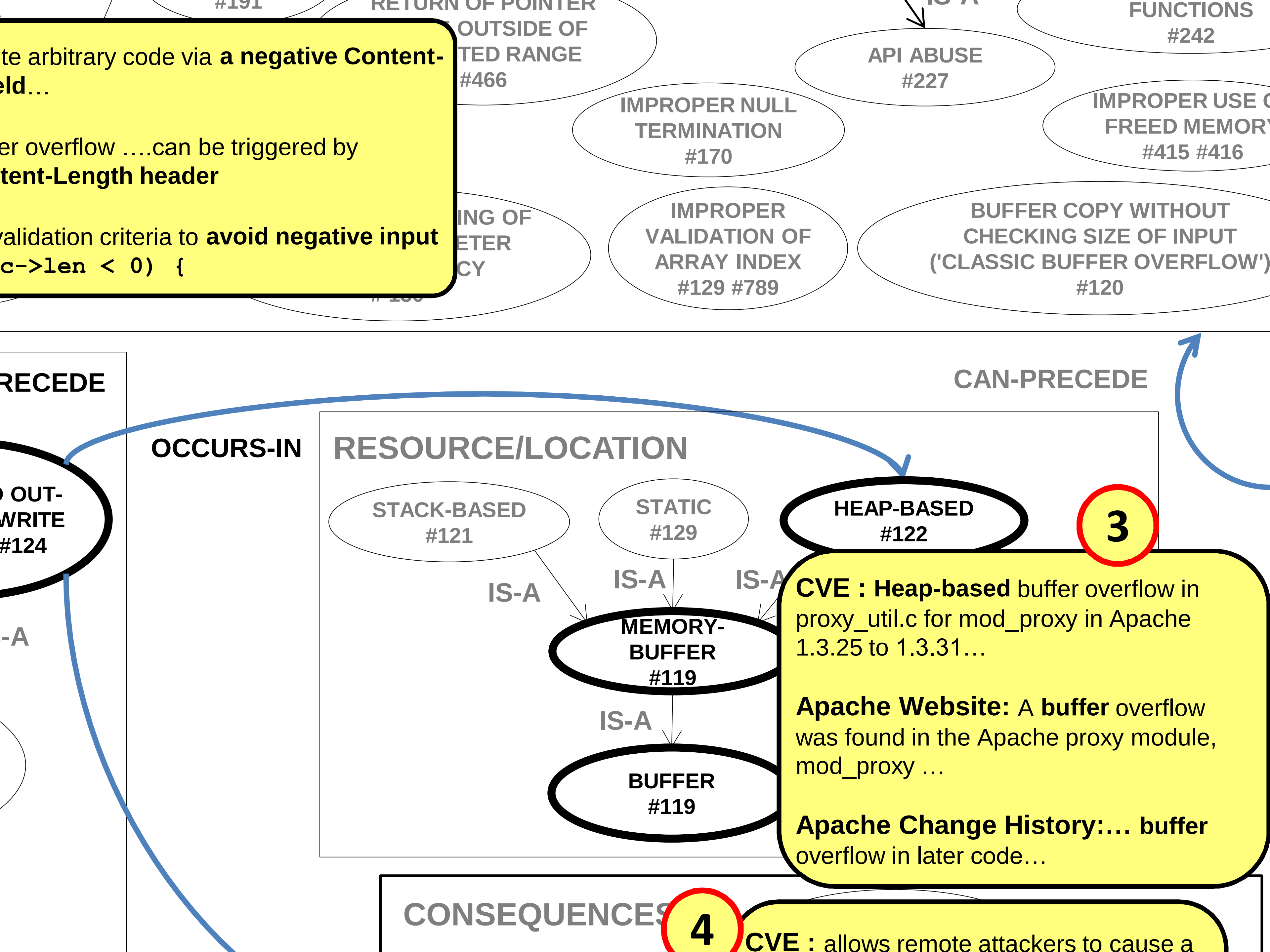
Apache Website: buffer overflow ….can be triggered by receiving an **invalid Content-Length header**

Source Code: Added validation criteria to **avoid negative input**

```
if (c->len < 0) {
```

CVE : …. causes **a large amount of data to be copied** …

Apache Website: This issue may lead to **remote arbitrary code execution** on some BSD platforms…


#191
RETURN OF POINTER OUTSIDE OF TED RANGE #466
IS-A
API ABUSE #227
FUNCTIONS #242
IMPROPER NULL TERMINATION #170
IMPROPER USE O FREED MEMORY #415 #416
ING OF ETER CY #130
IMPROPER VALIDATION OF ARRAY INDEX #129 #789
BUFFER COPY WITHOUT CHECKING SIZE OF INPUT ('CLASSIC BUFFER OVERFLOW') #120
RECEDE
CAN-PRECEDE
OCCURS-IN
RESOURCE/LOCATION
O OUT- WRITE #124
-A
STACK-BASED #121
STATIC #129
HEAP-BASED #122
IS-A
IS-A
IS-A
MEMORY-BUFFER #119
IS-A
BUFFER #119
3
CONSEQUENCES
4

te arbitrary code via **a negative Content-eld**…

er overflow ….can be triggered by **tent-Length header**

validation criteria to **avoid negative input**

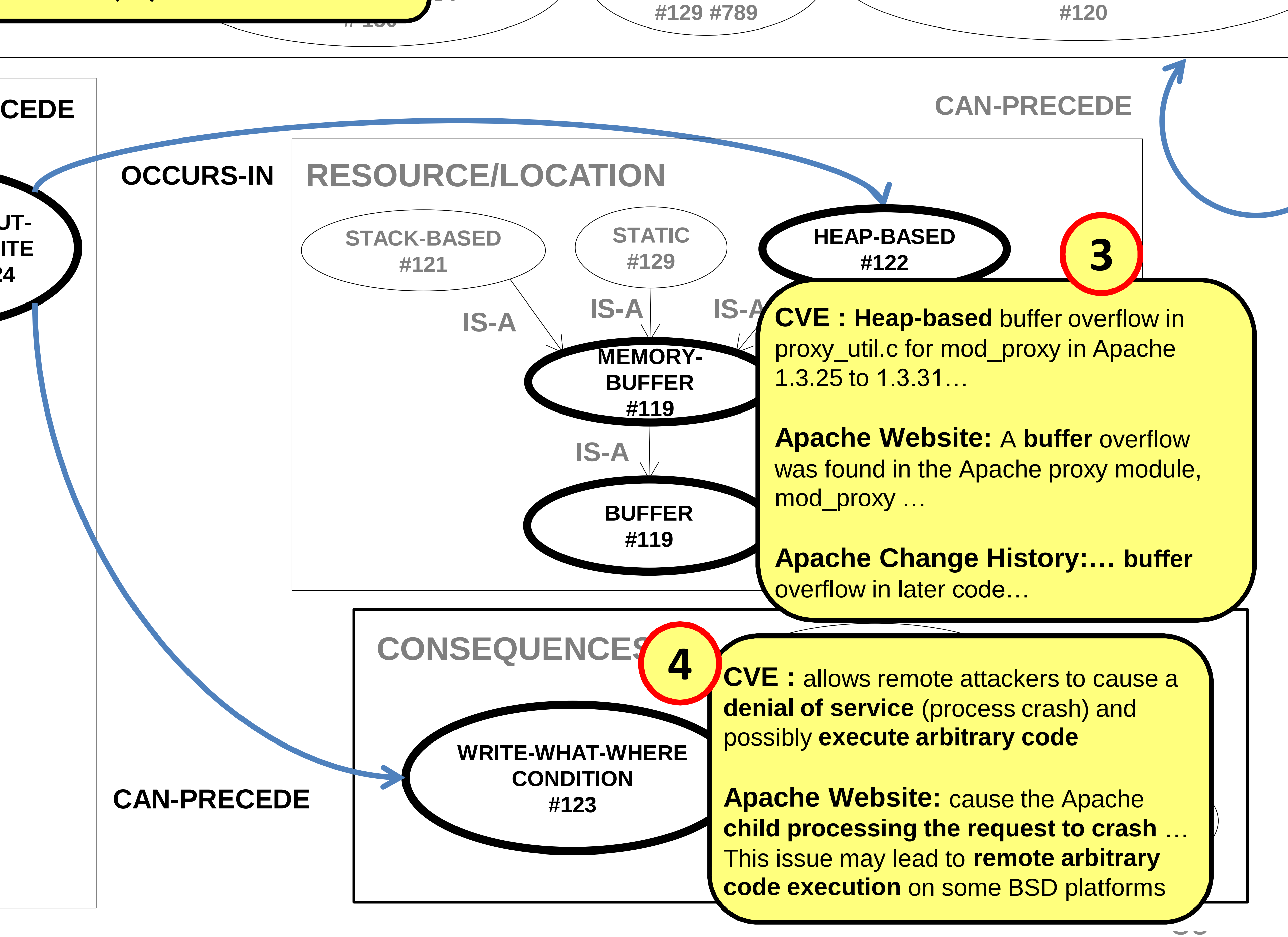
```
c->len < 0) {
```

CVE : Heap-based buffer overflow in proxy_util.c for mod_proxy in Apache 1.3.25 to 1.3.31…

Apache Website: A **buffer** overflow was found in the Apache proxy module, mod_proxy …

Apache Change History:… buffer overflow in later code…

CVE : allows remote attackers to cause a



CVE : Heap-based buffer overflow in proxy_util.c for mod_proxy in Apache 1.3.25 to 1.3.31…

Apache Website: A **buffer** overflow was found in the Apache proxy module, mod_proxy …

Apache Change History:… buffer overflow in later code…

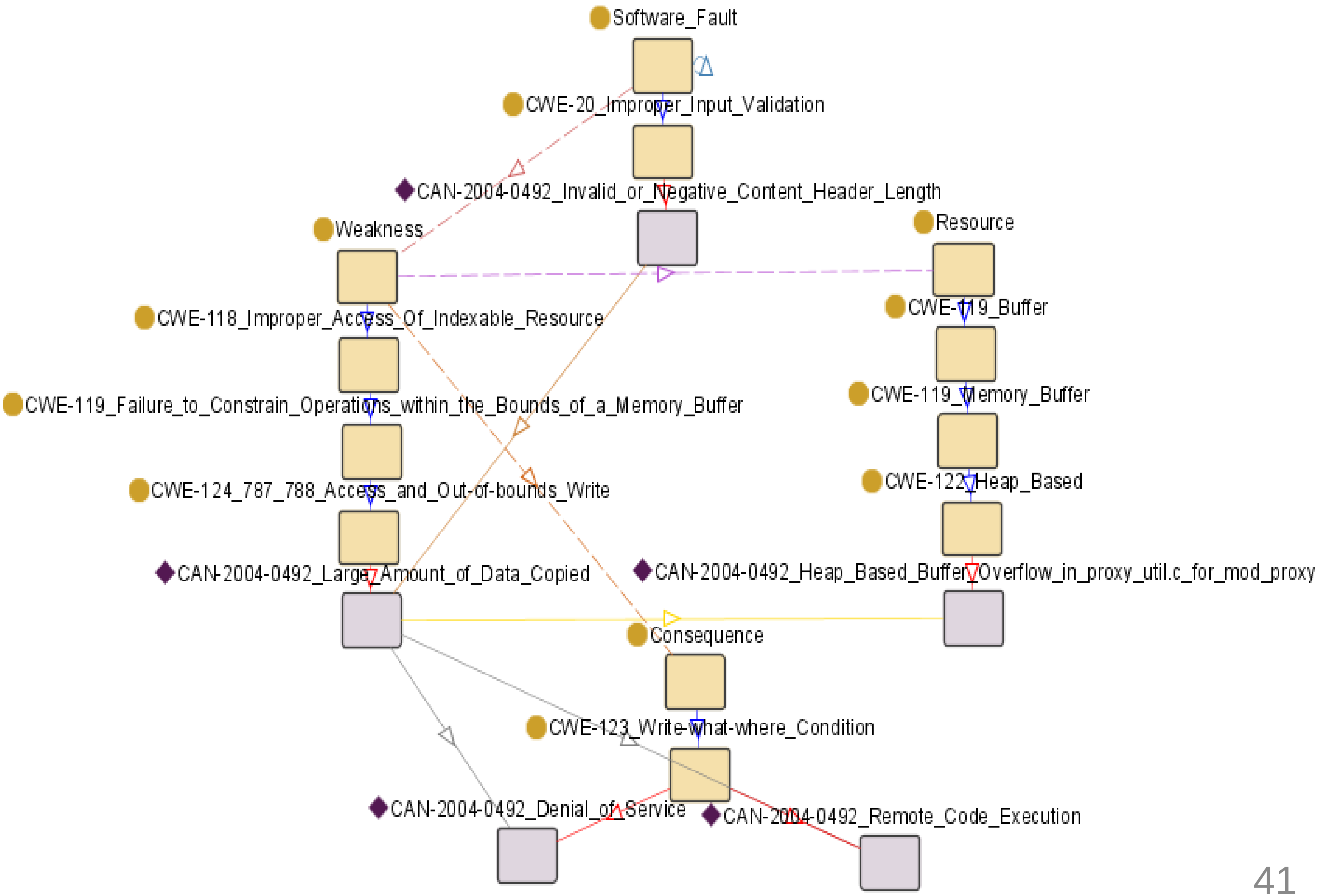
CVE : allows remote attackers to cause a **denial of service** (process crash) and possibly **execute arbitrary code**

Apache Website: cause the Apache **child processing the request to crash** … This issue may lead to **remote arbitrary code execution** on some BSD platforms

Ontology Representation

- Semantic web based representation
  - Allow inferences and queries over a large collection of semantically annotated vulnerability artifacts
  - Examples
    - “Show past vulnerabilities that related to **buffer overflow weaknesses** that precedes **arbitrary code execution**”
    - “Which **software fault** most often precedes the **buffer overflow weaknesses**?”

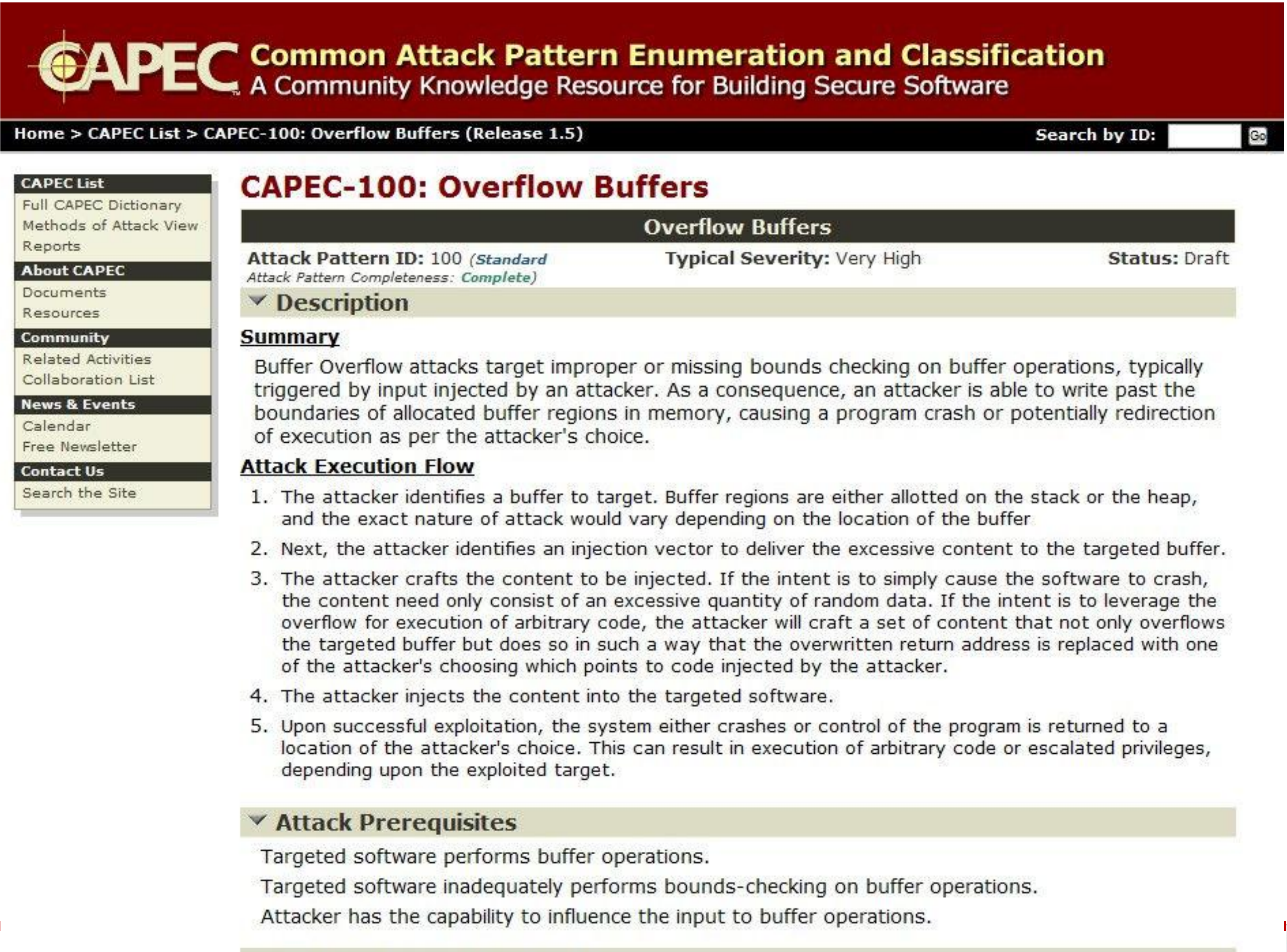
Semantic Web Visualization


Software_Fault
CWE-20_Improper_Input_Validation
CAN-2004-0492_Invalid_or_Negative_Content_Header_Length
Weakness
Resource
CWE-118_Improper_Access_Of_Indexable_Resource
CWE-119_Buffer
CWE-119_Failure_to_Constrain_Operations_within_the_Bounds_of_a_Memory_Buffer
CWE-119_Memory_Buffer
CWE-124_787_788_Access_and_Out-of-bounds_Write
CWE-122_Heap_Based
CAN-2004-0492_Large_Amount_of_Data_Copied
CAN-2004-0492_Heap_Based_Buffer_Overflow_in_proxy_util.c_for_mod_proxy
Consequence
CWE-123_Write-what-where_Condition
CAN-2004-0492_Denial_of_Service
CAN-2004-0492_Remote_Code_Execution

Common Attack Pattern Enumeration and Classification (**CAPEC**)

- A shared indexing standard for common attacks patterns used in exploits or malware
- Attack patterns
  - – Capture and communicate an attackers perspective
    - Common vocabulary to express attack vectors
  - – List of common methods to exploit vulnerabilities
  - – A “destructive” way of thinking
    - Know your enemy. Defense alone is not enough.
- http://capec.mitre.org/

CAPEC Example



CAPEC Common Attack Pattern Enumeration and Classification
A Community Knowledge Resource for Building Secure Software

Home > CAPEC List > CAPEC-100: Overflow Buffers (Release 1.5)

CAPEC-100: Overflow Buffers

Overflow Buffers

Attack Pattern ID: 100 (*Standard Attack Pattern Completeness: Complete*) **Typical Severity:** Very High **Status:** Draft

Description

Summary

Buffer Overflow attacks target improper or missing bounds checking on buffer operations, typically triggered by input injected by an attacker. As a consequence, an attacker is able to write past the boundaries of allocated buffer regions in memory, causing a program crash or potentially redirection of execution as per the attacker's choice.

Attack Execution Flow

1. The attacker identifies a buffer to target. Buffer regions are either allotted on the stack or the heap, and the exact nature of attack would vary depending on the location of the buffer
2. Next, the attacker identifies an injection vector to deliver the excessive content to the targeted buffer.
3. The attacker crafts the content to be injected. If the intent is to simply cause the software to crash, the content need only consist of an excessive quantity of random data. If the intent is to leverage the overflow for execution of arbitrary code, the attacker will craft a set of content that not only overflows the targeted buffer but does so in such a way that the overwritten return address is replaced with one of the attacker's choosing which points to code injected by the attacker.
4. The attacker injects the content into the targeted software.
5. Upon successful exploitation, the system either crashes or control of the program is returned to a location of the attacker's choice. This can result in execution of arbitrary code or escalated privileges, depending upon the exploited target.

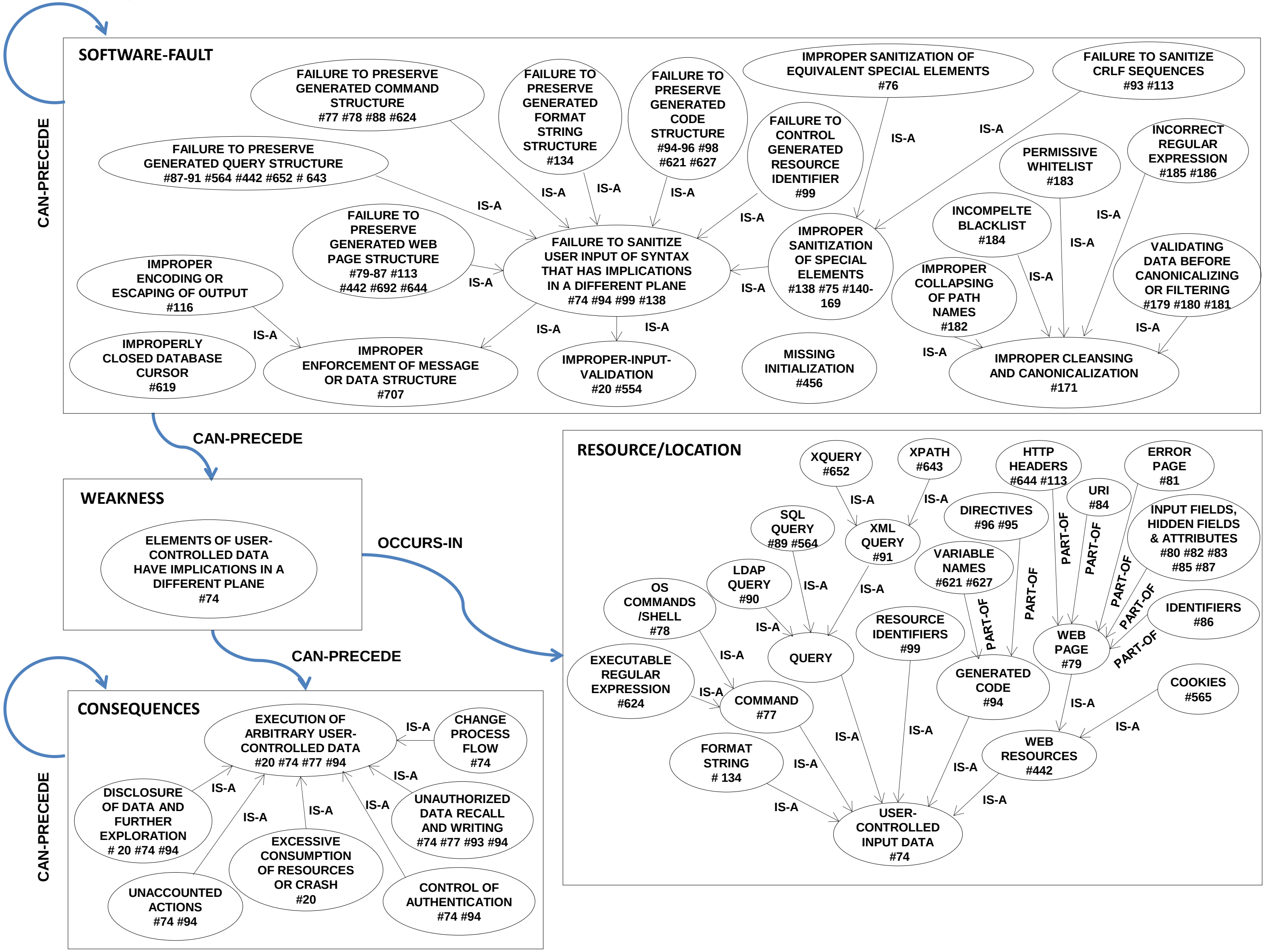
Attack Prerequisites

Targeted software performs buffer operations.

Targeted software inadequately performs bounds-checking on buffer operations.

Attacker has the capability to influence the input to buffer operations.

Ongoing Work (Injection Template)


SOFTWARE-FAULT
CAN-PRECEDE
FAILURE TO PRESERVE GENERATED COMMAND STRUCTURE #77 #78 #88 #624
FAILURE TO PRESERVE GENERATED FORMAT STRING STRUCTURE #134
FAILURE TO PRESERVE GENERATED CODE STRUCTURE #94-96 #98 #621 #627
IMPROPER SANITIZATION OF EQUIVALENT SPECIAL ELEMENTS #76
FAILURE TO SANITIZE CRLF SEQUENCES #93 #113
FAILURE TO PRESERVE GENERATED QUERY STRUCTURE #87-91 #564 #442 #652 # 643
FAILURE TO CONTROL GENERATED RESOURCE IDENTIFIER #99
PERMISSIVE WHITELIST #183
INCORRECT REGULAR EXPRESSION #185 #186
FAILURE TO PRESERVE GENERATED WEB PAGE STRUCTURE #79-87 #113 #442 #692 #644
FAILURE TO SANITIZE USER INPUT OF SYNTAX THAT HAS IMPLICATIONS IN A DIFFERENT PLANE #74 #94 #99 #138
IMPROPER SANITIZATION OF SPECIAL ELEMENTS #138 #75 #140-169
INCOMPELTE BLACKLIST #184
IMPROPER ENCODING OR ESCAPING OF OUTPUT #116
IMPROPER COLLAPSING OF PATH NAMES #182
VALIDATING DATA BEFORE CANONICALIZING OR FILTERING #179 #180 #181
IMPROPERLY CLOSED DATABASE CURSOR #619
IMPROPER ENFORCEMENT OF MESSAGE OR DATA STRUCTURE #707
IMPROPER-INPUT-VALIDATION #20 #554
MISSING INITIALIZATION #456
IMPROPER CLEANSING AND CANONICALIZATION #171
IS-A
CAN-PRECEDE
WEAKNESS
ELEMENTS OF USER-CONTROLLED DATA HAVE IMPLICATIONS IN A DIFFERENT PLANE #74
OCCURS-IN
CAN-PRECEDE
CONSEQUENCES
CAN-PRECEDE
EXECUTION OF ARBITRARY USER-CONTROLLED DATA #20 #74 #77 #94
CHANGE PROCESS FLOW #74
DISCLOSURE OF DATA AND FURTHER EXPLORATION # 20 #74 #94
UNAUTHORIZED DATA RECALL AND WRITING #74 #77 #93 #94
EXCESSIVE CONSUMPTION OF RESOURCES OR CRASH #20
UNACCOUNTED ACTIONS #74 #94
CONTROL OF AUTHENTICATION #74 #94
RESOURCE/LOCATION
XQUERY #652
XPATH #643
HTTP HEADERS #644 #113
ERROR PAGE #81
URI #84
SQL QUERY #89 #564
XML QUERY #91
DIRECTIVES #96 #95
INPUT FIELDS, HIDDEN FIELDS & ATTRIBUTES #80 #82 #83 #85 #87
LDAP QUERY #90
VARIABLE NAMES #621 #627
OS COMMANDS /SHELL #78
RESOURCE IDENTIFIERS #99
IDENTIFIERS #86
QUERY
WEB PAGE #79
EXECUTABLE REGULAR EXPRESSION #624
GENERATED CODE #94
COMMAND #77
COOKIES #565
FORMAT STRING # 134
WEB RESOURCES #442
USER-CONTROLLED INPUT DATA #74
IS-A
PART-OF

Ongoing Work (Injection Template)

Apache Website CVE-2007-5000 : ….a cross-site scripting attack is possible….

NVD CVE-2007-5000: a Cross-site scripting (XSS) vulnerability in the ….mod_imap module….

Source Code Repository developer fix documentation: Fix cross-site-scripting issue by escaping the URI…

Source Code Repository Code Difference:

```
File: mod_imagemap.c
Line 485 and 490 modified to
escape html in URI:
ap_escape_html(r->pool, r->uri)
```

CAPEC-63: Simple Script Injection:

(Experimentation) Use a list of XSS probe strings to inject script into resources accessed by the application

(Exploit) Develop malicious JavaScript that is injected through vectors identified during the Experiment Phase

Source Code Repository developer fix documentation: ….ensure that a charset parameter is sent in the content-type …

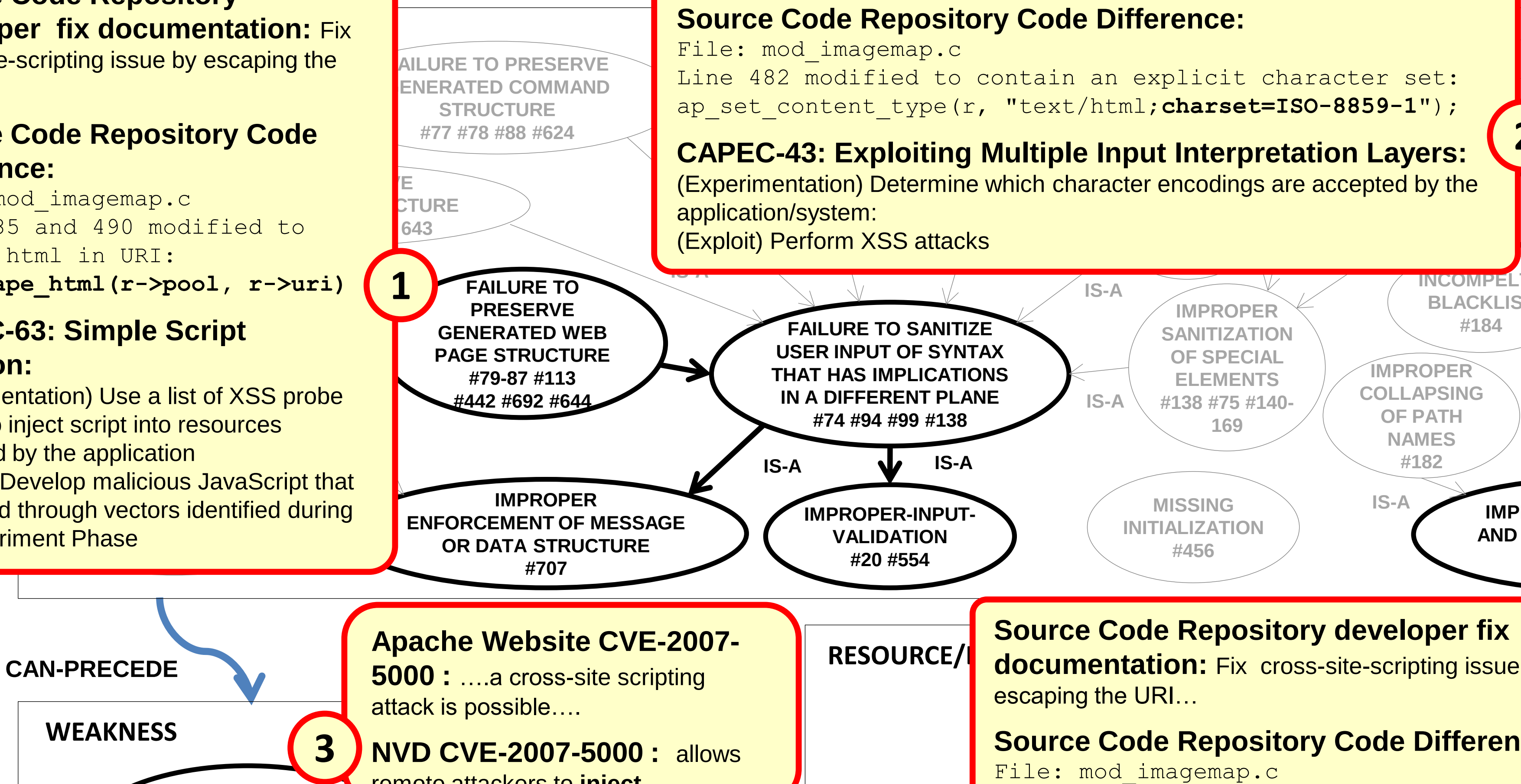
Source Code Repository Code Difference:

```
File: mod_imagemap.c
Line 482 modified to contain an explicit character set:
ap_set_content_type(r, "text/html;charset=ISO-8859-1");
```

CAPEC-43: Exploiting Multiple Input Interpretation Layers:

(Experimentation) Determine which character encodings are accepted by the application/system:

(Exploit) Perform XSS attacks



AILURE TO PRESERVE ENERATED COMMAND STRUCTURE #77 #78 #88 #624
E CTURE 643
1
FAILURE TO PRESERVE GENERATED WEB PAGE STRUCTURE #79-87 #113 #442 #692 #644
FAILURE TO SANITIZE USER INPUT OF SYNTAX THAT HAS IMPLICATIONS IN A DIFFERENT PLANE #74 #94 #99 #138
IS-A
IS-A
IMPROPER SANITIZATION OF SPECIAL ELEMENTS #138 #75 #140-169
INCOMPLE BLACKLIS #184
IMPROPER COLLAPSING OF PATH NAMES #182
IS-A
IS-A
IS-A
IMPROPER ENFORCEMENT OF MESSAGE OR DATA STRUCTURE #707
IMPROPER-INPUT-VALIDATION #20 #554
MISSING INITIALIZATION #456
IS-A
IMP AND
2
CAN-PRECEDE
WEAKNESS
RESOURCE/
3

Apache Website CVE-2007-5000 : ….a cross-site scripting attack is possible….

NVD CVE-2007-5000 : allows remote attackers to **inject**

Source Code Repository developer fix documentation: Fix cross-site-scripting issue escaping the URI…

Source Code Repository Code Differen

```
File: mod_imagemap.c
```

Ongoing Work (Injection Template)

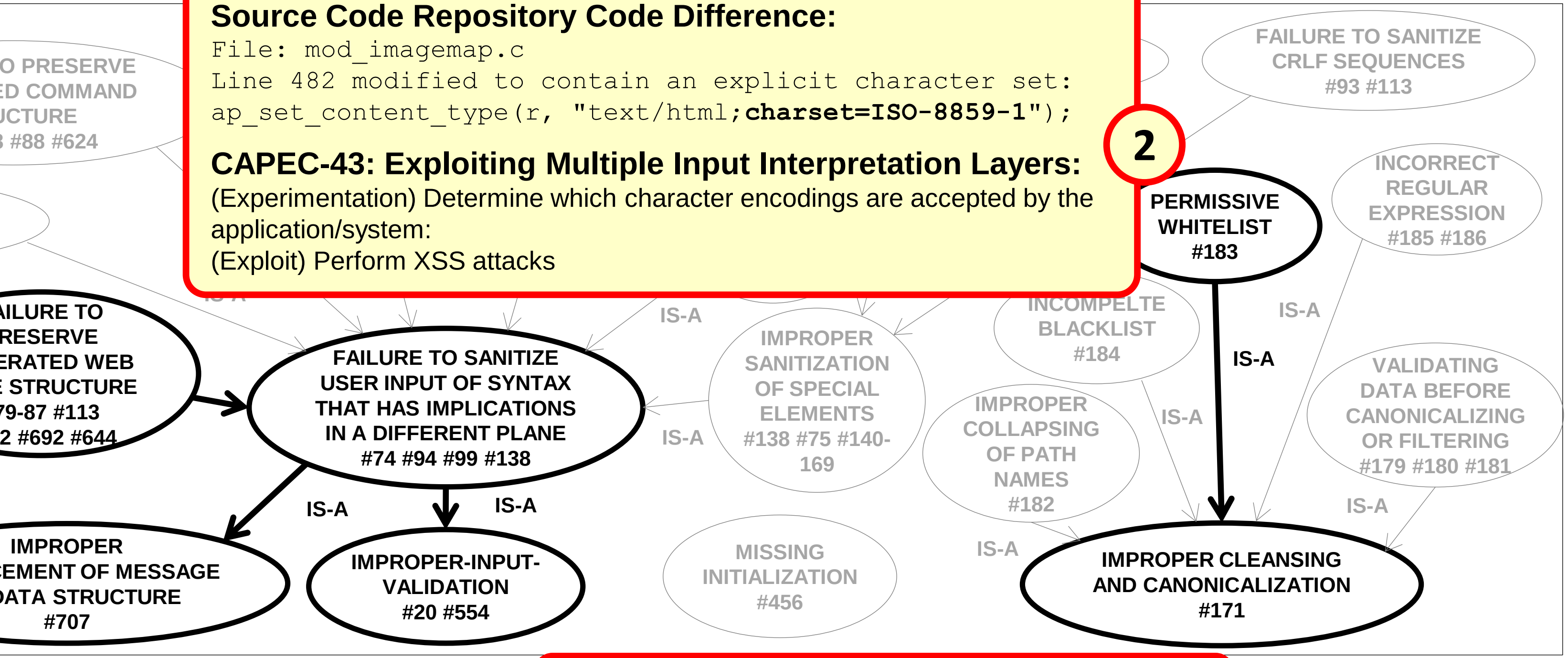
Source Code Repository developer fix documentation:
….ensure that a charset parameter is sent in the content-type …

Source Code Repository Code Difference:

```
File: mod_imagemap.c
Line 482 modified to contain an explicit character set:
ap_set_content_type(r, "text/html;charset=ISO-8859-1");
```

CAPEC-43: Exploiting Multiple Input Interpretation Layers:
(Experimentation) Determine which character encodings are accepted by the application/system:
(Exploit) Perform XSS attacks

2



Website CVE-2007-
.a cross-site scripting
ossible….

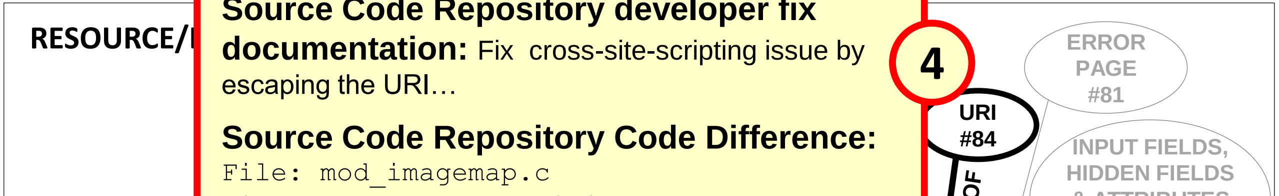
E-2007-5000 : allows
ackers to **inject**…

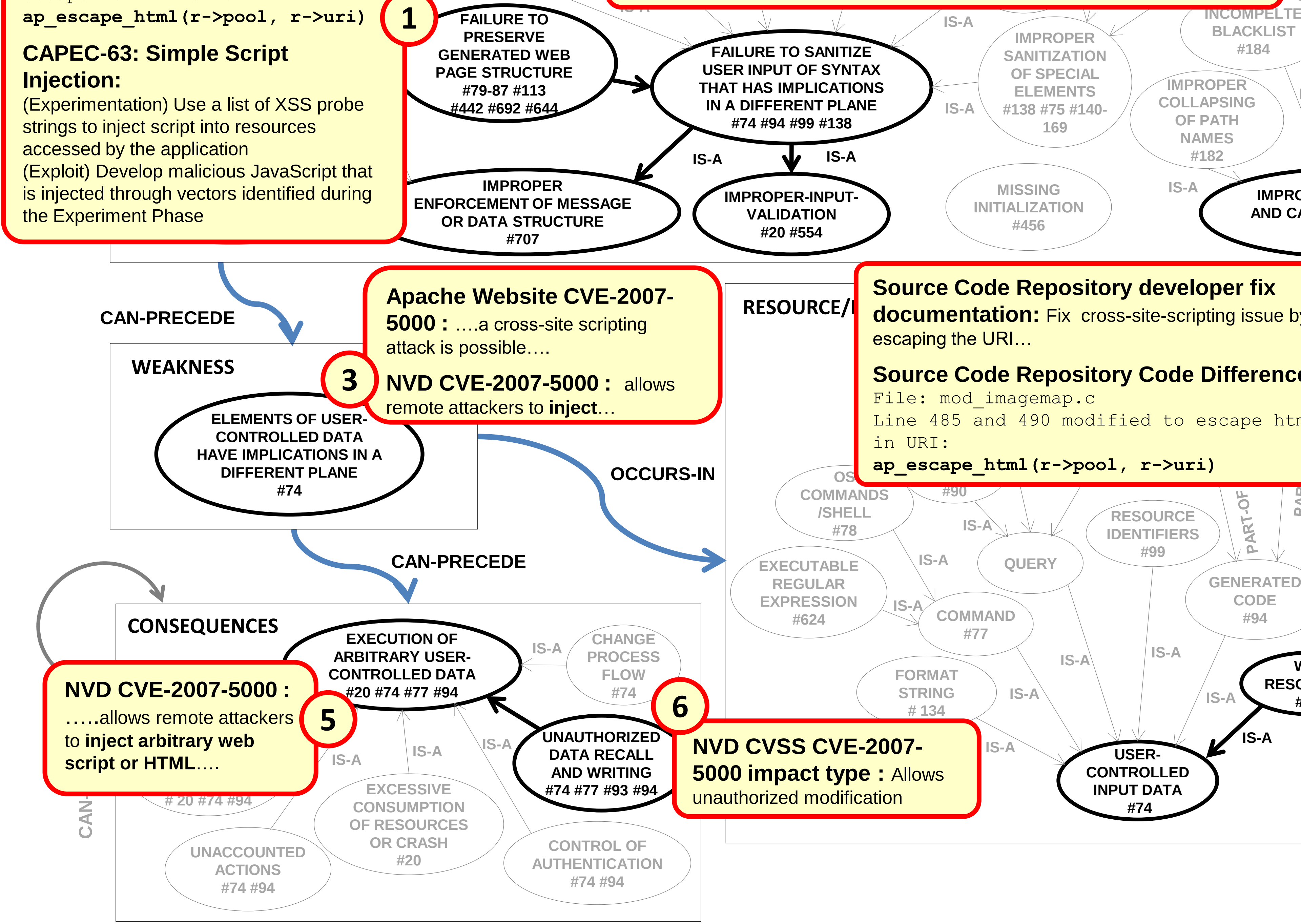
Source Code Repository developer fix documentation: Fix cross-site-scripting issue by escaping the URI…

Source Code Repository Code Difference:

```
File: mod_imagemap.c
```

4





```
ap_escape_html(r->pool, r->uri)
```

CAPEC-63: Simple Script Injection:
(Experimentation) Use a list of XSS probe strings to inject script into resources accessed by the application
(Exploit) Develop malicious JavaScript that is injected through vectors identified during the Experiment Phase

1

FAILURE TO PRESERVE GENERATED WEB PAGE STRUCTURE #79-87 #113 #442 #692 #644

FAILURE TO SANITIZE USER INPUT OF SYNTAX THAT HAS IMPLICATIONS IN A DIFFERENT PLANE #74 #94 #99 #138

IS-A

IS-A

IMPROPER ENFORCEMENT OF MESSAGE OR DATA STRUCTURE #707

IMPROPER-INPUT-VALIDATION #20 #554

IS-A

IS-A

IMPROPER SANITIZATION OF SPECIAL ELEMENTS #138 #75 #140-169

INCOMPLETE BLACKLIST #184

IMPROPER COLLAPSING OF PATH NAMES #182

MISSING INITIALIZATION #456

IS-A

IMPRO AND CA

CAN-PRECEDE

WEAKNESS

ELEMENTS OF USER-CONTROLLED DATA HAVE IMPLICATIONS IN A DIFFERENT PLANE #74

3

Apache Website CVE-2007-5000 : ….a cross-site scripting attack is possible….

NVD CVE-2007-5000 : allows remote attackers to **inject**…

OCCURS-IN

CAN-PRECEDE

RESOURCE/I

Source Code Repository developer fix documentation: Fix cross-site-scripting issue by escaping the URI…

Source Code Repository Code Differenc

```
File: mod_imagemap.c
Line 485 and 490 modified to escape htm
in URI:
ap_escape_html(r->pool, r->uri)
```

OS COMMANDS /SHELL #78

#90

IS-A

QUERY

RESOURCE IDENTIFIERS #99

PART-OF

PAR

EXECUTABLE REGULAR EXPRESSION #624

IS-A

IS-A

COMMAND #77

GENERATED CODE #94

FORMAT STRING # 134

IS-A

IS-A

IS-A

IS-A

IS-A

IS-A

USER-CONTROLLED INPUT DATA #74

W RESO #

CONSEQUENCES

EXECUTION OF ARBITRARY USER-CONTROLLED DATA #20 #74 #77 #94

IS-A

CHANGE PROCESS FLOW #74

NVD CVE-2007-5000 : …..allows remote attackers to **inject arbitrary web script or HTML**….

5

IS-A

IS-A

IS-A

UNAUTHORIZED DATA RECALL AND WRITING #74 #77 #93 #94

6

NVD CVSS CVE-2007-5000 impact type : Allows unauthorized modification

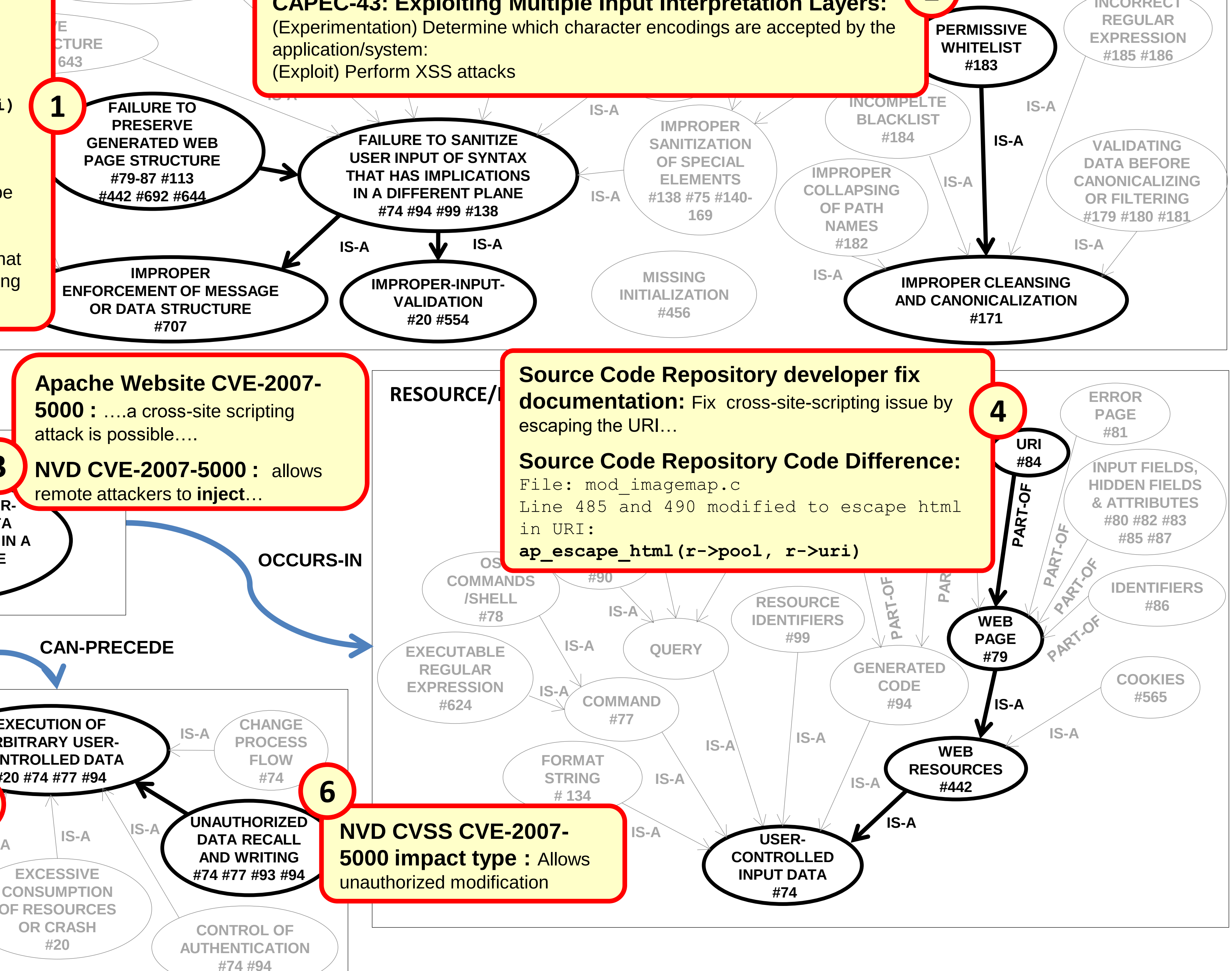
# 20 #74 #94

CAN-

EXCESSIVE CONSUMPTION OF RESOURCES OR CRASH #20

UNACCOUNTED ACTIONS #74 #94

CONTROL OF AUTHENTICATION #74 #94


E
CTURE
643
i)
e
hat
ng
1
FAILURE TO PRESERVE GENERATED WEB PAGE STRUCTURE #79-87 #113 #442 #692 #644
FAILURE TO SANITIZE USER INPUT OF SYNTAX THAT HAS IMPLICATIONS IN A DIFFERENT PLANE #74 #94 #99 #138
IS-A
IMPROPER ENFORCEMENT OF MESSAGE OR DATA STRUCTURE #707
IS-A
IMPROPER-INPUT-VALIDATION #20 #554
IS-A
IMPROPER SANITIZATION OF SPECIAL ELEMENTS #138 #75 #140-169
IS-A
MISSING INITIALIZATION #456
IMPROPER COLLAPSING OF PATH NAMES #182
INCOMPELTE BLACKLIST #184
IS-A
PERMISSIVE WHITELIST #183
IS-A
INCORRECT REGULAR EXPRESSION #185 #186
IS-A
VALIDATING DATA BEFORE CANONICALIZING OR FILTERING #179 #180 #181
IS-A
IS-A
IS-A
IMPROPER CLEANSING AND CANONICALIZATION #171
3
R-
A
IN A
E
OCCURS-IN
CAN-PRECEDE
RESOURCE/I
4
ERROR PAGE #81
URI #84
PART-OF
INPUT FIELDS, HIDDEN FIELDS & ATTRIBUTES #80 #82 #83 #85 #87
PART-OF
PART-OF
IDENTIFIERS #86
PART-OF
PART-OF
PAR
WEB PAGE #79
IS-A
COOKIES #565
IS-A
WEB RESOURCES #442
IS-A
OS COMMANDS /SHELL #78
#90
IS-A
QUERY
RESOURCE IDENTIFIERS #99
GENERATED CODE #94
EXECUTABLE REGULAR EXPRESSION #624
IS-A
IS-A
COMMAND #77
FORMAT STRING # 134
IS-A
IS-A
IS-A
IS-A
IS-A
USER-CONTROLLED INPUT DATA #74
EXECUTION OF RBITRARY USER-NTROLLED DATA 20 #74 #77 #94
IS-A
CHANGE PROCESS FLOW #74
6
UNAUTHORIZED DATA RECALL AND WRITING #74 #77 #93 #94
IS-A
IS-A
A
EXCESSIVE CONSUMPTION OF RESOURCES OR CRASH #20
CONTROL OF AUTHENTICATION #74 #94

CAPEC-43: Exploiting Multiple Input Interpretation Layers:
(Experimentation) Determine which character encodings are accepted by the application/system:
(Exploit) Perform XSS attacks

Apache Website CVE-2007-5000 : ….a cross-site scripting attack is possible….

NVD CVE-2007-5000 : allows remote attackers to **inject**…

Source Code Repository developer fix documentation: Fix cross-site-scripting issue by escaping the URI…

Source Code Repository Code Difference:

```
File: mod_imagemap.c
Line 485 and 490 modified to escape html
in URI:
ap_escape_html(r->pool, r->uri)
```

NVD CVSS CVE-2007-5000 impact type : Allows unauthorized modification

Future Work

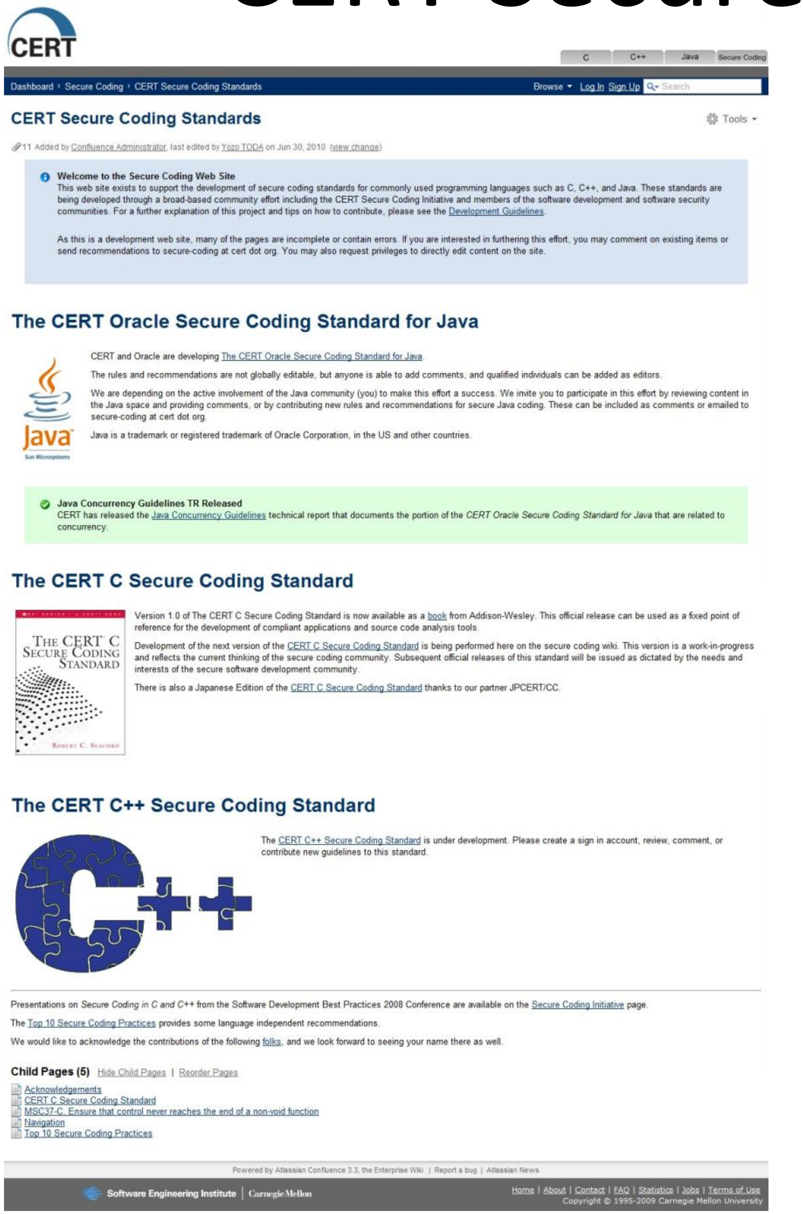
- Integrate with existing static and dynamic analysis tools to enhance reporting capabilities
  - Provide layers of guidance to a developer upon detection of a software flaw
  - Organize and retrieve knowledge of past vulnerabilities
  - Verify patch submissions
- Investigate project/developer specific coding errors and vulnerability fix patterns
- Other usage scenarios in the SDLC

Some take aways…

- Ask Johnny (or your software vendor):
  - *How many CWEs have you attempted to explicitly avoid in your software?*
  - *What CWEs can our Threats take advantage of?*
    - *I want you to build a shopping cart, while avoiding those CWEs…*
  - *What CAPECs do your testing efforts map to?*
  - *What CWEs do the vulnerabilities in your code typically map to? Have you taken any training for them?*
    - *Have you looked at the semantic templates by being developed at UNO/NUCIA for those CWEs?*
      - *http://faculty.ist.unomaha.edu/rgandhi/st/*

CERT Secure Coding Guidelines

https://www.securecoding.cert.org/



CERT Secure Coding Standards

11 Added by Confluence Administrator, last edited by Yozo TODA on Jun 30, 2010 (view change)

Welcome to the Secure Coding Web Site
This web site exists to support the development of secure coding standards for commonly used programming languages such as C, C++, and Java. These standards are being developed through a broad-based community effort including the CERT Secure Coding Initiative and members of the software development and software security communities. For a further explanation of this project and tips on how to contribute, please see the Development Guidelines.

As this is a development web site, many of the pages are incomplete or contain errors. If you are interested in furthering this effort, you may comment on existing items or send recommendations to secure-coding at cert dot org. You may also request privileges to directly edit content on the site.

The CERT Oracle Secure Coding Standard for Java

CERT and Oracle are developing The CERT Oracle Secure Coding Standard for Java.

The rules and recommendations are not globally editable, but anyone is able to add comments, and qualified individuals can be added as editors.

We are depending on the active involvement of the Java community (you) to make this effort a success. We invite you to participate in this effort by reviewing content in the Java space and providing comments, or by contributing new rules and recommendations for secure Java coding. These can be included as comments or emailed to secure-coding at cert dot org.

Java is a trademark or registered trademark of Oracle Corporation, in the US and other countries.

Java Concurrency Guidelines TR Released
CERT has released the Java Concurrency Guidelines technical report that documents the portion of the *CERT Oracle Secure Coding Standard for Java* that are related to concurrency.

The CERT C Secure Coding Standard

Version 1.0 of The CERT C Secure Coding Standard is now available as a book from Addison-Wesley. This official release can be used as a fixed point of reference for the development of compliant applications and source code analysis tools.

Development of the next version of the CERT C Secure Coding Standard is being performed here on the secure coding wiki. This version is a work-in-progress and reflects the current thinking of the secure coding community. Subsequent official releases of this standard will be issued as dictated by the needs and interests of the secure software development community.

There is also a Japanese Edition of the CERT C Secure Coding Standard thanks to our partner JPCERT/CC.

The CERT C++ Secure Coding Standard

The CERT C++ Secure Coding Standard is under development. Please create a sign in account, review, comment, or contribute new guidelines to this standard.

Presentations on *Secure Coding in C and C++* from the Software Development Best Practices 2008 Conference are available on the Secure Coding Initiative page.

The Top 10 Secure Coding Practices provides some language independent recommendations.

We would like to acknowledge the contributions of the following folks, and we look forward to seeing your name there as well.

Child Pages (5) Hide Child Pages | Reorder Pages

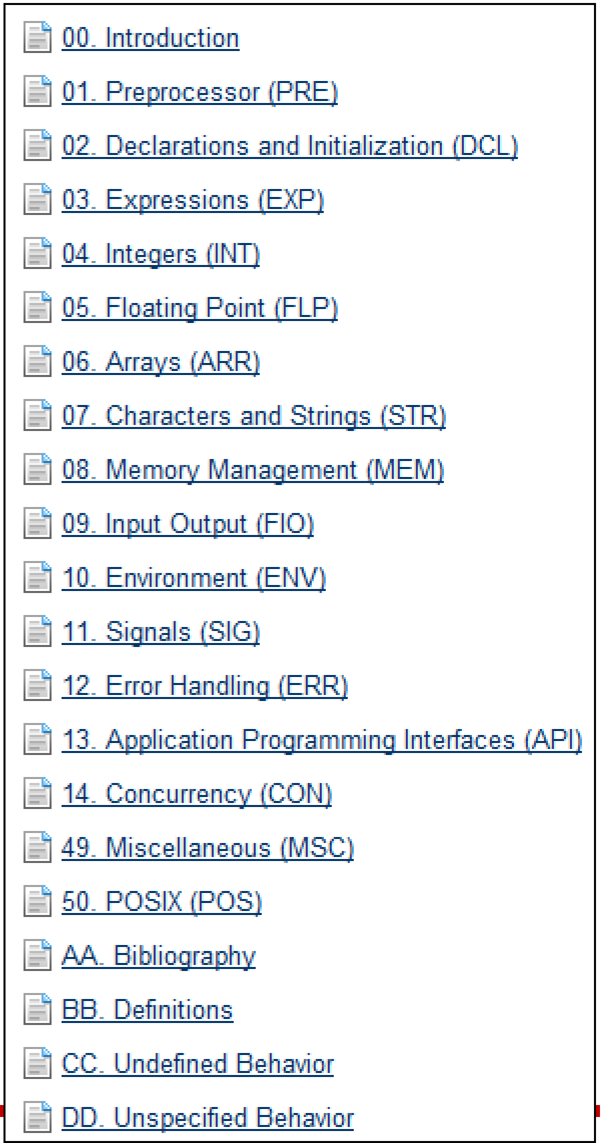
- Acknowledgements
- CERT C Secure Coding Standard
- MSC37-C. Ensure that control never reaches the end of a non-void function
- Navigation
- Top 10 Secure Coding Practices

Powered by Atlassian Confluence 3.3, the Enterprise Wiki | Report a bug | Atlassian News

Software Engineering Institute | Carnegie Mellon

Home | About | Contact | FAQ | Statistics | Jobs | Terms of Use
Copyright © 1995-2009 Carnegie Mellon University

CERT C Secure Coding Standard

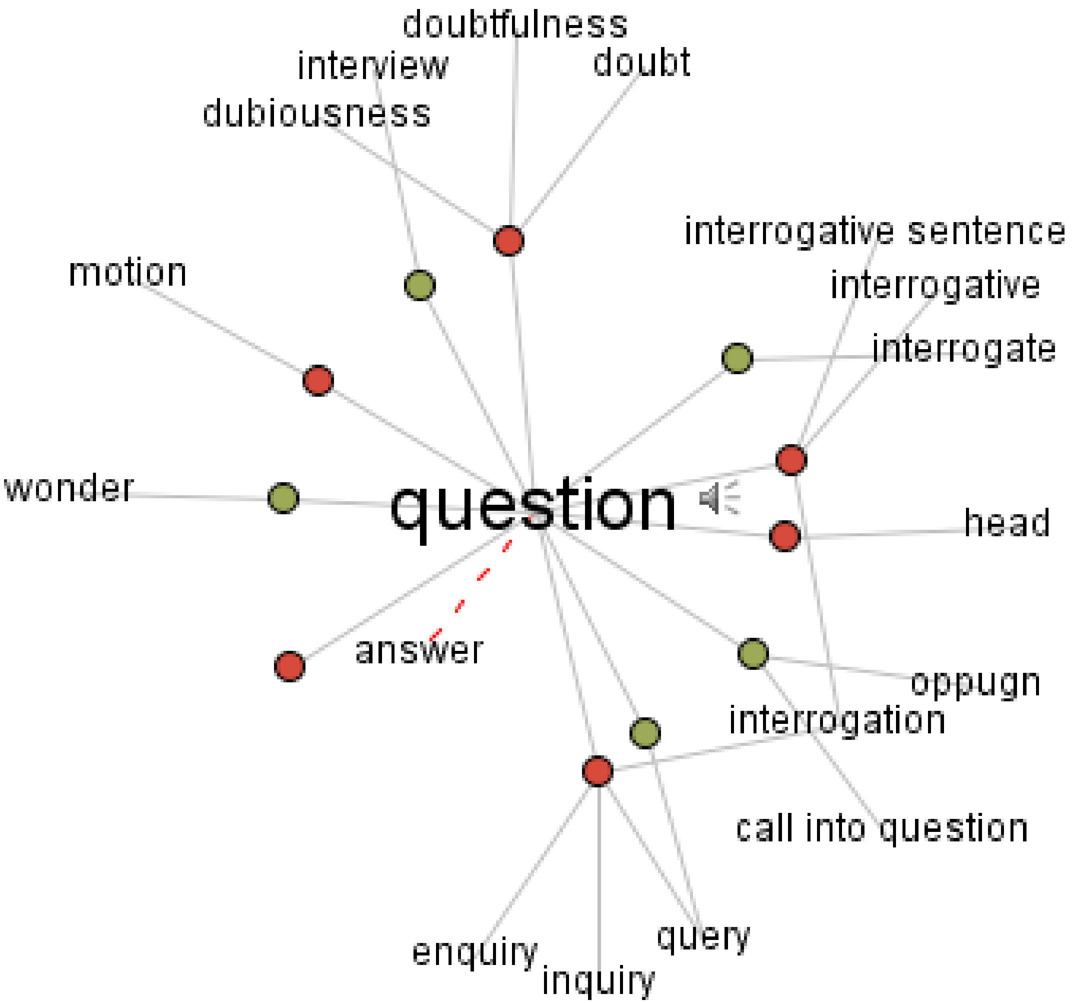


- 00. Introduction
- 01. Preprocessor (PRE)
- 02. Declarations and Initialization (DCL)
- 03. Expressions (EXP)
- 04. Integers (INT)
- 05. Floating Point (FLP)
- 06. Arrays (ARR)
- 07. Characters and Strings (STR)
- 08. Memory Management (MEM)
- 09. Input Output (FIO)
- 10. Environment (ENV)
- 11. Signals (SIG)
- 12. Error Handling (ERR)
- 13. Application Programming Interfaces (API)
- 14. Concurrency (CON)
- 49. Miscellaneous (MSC)
- 50. POSIX (POS)
- AA. Bibliography
- BB. Definitions
- CC. Undefined Behavior
- DD. Unspecified Behavior

Acknowledgement

- This research is funded in part by Department of Defense (DoD)/Air Force Office of Scientific Research (AFOSR), NSF Award Number FA9550-07-1-0499, under the title “**High Assurance Software**”

Thank you for your Attention


doubtfulness
interview
doubt
dubiousness
interrogative sentence
motion
interrogative
interrogate
wonder
question
head
answer
oppugn
interrogation
call into question
enquiry
query
inquiry