

# CWE User Experience Working Group Meeting

---

February 26, 2025



CWE is sponsored by [U.S. Department of Homeland Security](#) (DHS) [Cybersecurity and Infrastructure Security Agency](#) (CISA). Copyright © 1999–2024, [The MITRE Corporation](#). CWE and the CWE logo are trademarks of The MITRE Corporation.

# Agenda

---

**This meeting is being recorded :-)**

- **Primary topics**
  - Potential Macro-Usability Changes for CWE 4.17
  - Open Discussion
- **Reminders and Adjourn**



# Topic 1

## Potential Macro-Usability Changes for CWE 4.17

*Steve Christey Coley*



# Macro-Usability – Potential Changes for CWE 4.17

- **Previous work has been focused on front page and related menus**
- **Additional changes are possible**
  - only small technical changes can be made due to limited time
  - Features like “dynamic hierarchy navigator,” ability to change order of fields in an entry, ATT&CK-style matrix, etc. are not feasible for CWE 4.17 release
- **Goal: improve UX for first-time visitors to individual CWE entry pages**
  - Many visitors click on a CWE link from another web site and are driven to the individual web page for that CWE
- **Goal: make useful features more readily apparent**
  - The CWE site has lots of useful information that users probably don’t even notice
    - Hidden in menus, obscured in a link in the corner, etc.



# Improving UX for Individual Web Pages

- **Improve UX for people who first visit individual CWE entry pages**
  - Current default is “Comprehensive” – all fields (wall of text)
  - Change default to include only some elements
    - Maybe similar to “Conceptual” presentation
- **Change color/presentation for some kinds of entries (deprecated, obsolete, categories, views)**
  - Categories have pale yellow background but still look-and-feel the same as weaknesses
  - Deprecated/Obsolete look like regular active entries – could gray them out?
- **Change layout of complex fields (reduce wall-of-text)**
  - Currently it’s “all-or-nothing”
  - Default “Minimize” some complex/extensive fields
- **Remove/minimize IDs in category/views**
- **Presentation filters to select fields to show**
  - Big buttons that “blend in” to the page AND take up lots of space



# Example – CWE-79, Part 1

## CWE-79: Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting')

Lots of vertical space taken

Weakness ID: 79

Vulnerability Mapping: ALLOWED

Abstraction: Base

5 big buttons that take up lots of space

View customized information:

Conceptual

Operational

Mapping Friendly

Complete

Custom

“Complete” default

### ▼ Description

The product does not neutralize or incorrectly neutralizes user-controllable input before it is placed in output that is used as a web page that is served to other users.

### ► Extended Description

### ▼ Alternate Terms

**XSS:**

A common abbreviation for Cross-Site Scripting.

**HTML Injection:**

Used as a synonym of stored (Type 2) XSS.

**CSS:**

In the early years after initial discovery of XSS, "CSS" was a commonly-used acronym. However, this would cause confusion with "Cascading Style Sheets," so usage of this acronym has declined significantly.

(Missing diagram – a content issue)

Extended description could be hidden by default



# CWE-79, Part 2

## ▼ Common Consequences

Scope	Impact	Likelihood
Access Control Confidentiality	<b>Technical Impact:</b> <i>Bypass Protection Mechanism; Read Application Data</i>  The most common attack performed with cross-site scripting involves the disclosure of information stored in user cookies. Typically, a malicious user will craft a client-side script, which -- when parsed by a web browser -- performs some activity (such as sending all site cookies to a given E-mail address). This script will be loaded and run by each user visiting the web site. Since the site requesting to run the script has access to the cookies in question, the malicious script does also.	
Integrity Confidentiality Availability	<b>Technical Impact:</b> <i>Execute Unauthorized Code or Commands</i>  In some circumstances it may be possible to run arbitrary code on a victim's computer when cross-site scripting is combined with other flaws.	
Confidentiality Integrity Availability Access Control	<b>Technical Impact:</b> <i>Execute Unauthorized Code or Commands; Bypass Protection Mechanism; Read Application Data</i>  The consequences of cross-site scripting are varied and can be severe. The attack can cause a variety of problems for the end user that range in severity from an annoyance to complete account compromise. Some cross-site scripting vulnerabilities can be exploited to manipulate or steal cookies, create sessions that can be mistaken for those of a valid user, compromise confidential information, or execute malicious code on the end user systems for a nefarious purposes. Other damaging attacks include the disclosure of files, installation of Trojan horse programs, redirecting the user to some other page or site, running "Active X" controls (under Microsoft Internet Explorer) from sites that a user perceives as trustworthy, and modifying presentation of content.	

WALL.

OF.

TEXT.

Consider moving Scope to the right

No clear lines between rows / cells – maybe do alternate shading?

... but still laid out cleanly if somebody really needs to understand how bad it can be

“Likelihood” is rarely filled (and extremely context-dependent). Reduces horizontal space (and makes the field vertically longer)



# CWE-79, Part 3

## ▼ Potential Mitigations

### Phase: Architecture and Design

#### Strategy: Libraries or Frameworks

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

Examples of libraries and frameworks that make it easier to generate properly encoded output include Microsoft's Anti-XSS library, the OWASP ESAPI Encoding module, and Apache Wicket.

### Phases: Implementation; Architecture and Design

Understand the context in which your data will be used and the encoding that will be expected. This is especially important when transmitting data between different components, or when generating outputs that can contain multiple encodings at the same time, such as web pages or multi-part mail messages. Study all expected communication protocols and data representations to determine the required encoding strategies.

For any data that will be output to another web page, especially any data that was received from external inputs, use the appropriate encoding on all non-alphanumeric characters.

Parts of the same output document may require different encodings, which will vary depending on whether the output is in the:

- HTML body
- Element attributes (such as `src="XYZ"`)
- URIs
- JavaScript sections
- Cascading Style Sheets and style property

etc. Note that HTML Entity Encoding is only appropriate for the HTML body.

Consult the XSS Prevention Cheat Sheet [[REF-724](#)] for more details on the types of encoding and escaping that are needed.

### Phases: Architecture and Design; Implementation

WALL.

OF.

TEXT.

*Layout is inconsistent with Common Consequences (which is more table-oriented)*

**... but a nicely organized table if you're tasked with fixing the thing**





# CWE-79, Part 4

## Relationships

Repetitive  
"ParentOf"

Change  
abstracti  
on  
icons?

Maybe not  
all views  
need to be  
expanded  
by default?


### Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf		74	<a href="#">Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection')</a>
ParentOf		80	<a href="#">Improper Neutralization of Script-Related HTML Tags in a Web Page (Basic XSS)</a>
ParentOf		81	<a href="#">Improper Neutralization of Script in an Error Message Web Page</a>
ParentOf		83	<a href="#">Improper Neutralization of Script in Attributes in a Web Page</a>
ParentOf		84	<a href="#">Improper Neutralization of Encoded URI Schemes in a Web Page</a>
ParentOf		85	<a href="#">Doubled Character XSS Manipulations</a>
ParentOf		86	<a href="#">Improper Neutralization of Invalid Characters in Identifiers in Web Pages</a>
ParentOf		87	<a href="#">Improper Neutralization of Alternate XSS Syntax</a>
PeerOf		352	<a href="#">Cross-Site Request Forgery (CSRF)</a>
CanFollow		113	<a href="#">Improper Neutralization of CRLF Sequences in HTTP Headers ('HTTP Request/Response Splitting')</a>
CanFollow		184	<a href="#">Incomplete List of Disallowed Inputs</a>
CanPrecede		494	<a href="#">Download of Code Without Integrity Check</a>

So many links with purple text

(But the  
icons  
are  
helpful  
to  
power  
users)

### Relevant to the view "Software Development" (CWE-699)

Nature	Type	ID	Name
MemberOf		137	<a href="#">Data Neutralization Issues</a>

Views have CWE IDs  
listed

### Relevant to the view "Weaknesses for Simplified Mapping of Published Vulnerabilities" (CWE-1003)

Nature	Type	ID	Name
ChildOf		74	<a href="#">Improper Neutralization of Special Elements in Output Used by a Downstream Component ('Injection')</a>



# CWE-79, Part 5

## ▼ References

WALL.

[REF-709] Jeremiah Grossman, Robert "RSnake" Hansen, Petko "pdp" D. Petkov, Anton Rager and Seth Fogie. "XSS Attacks". Syngress. 2007.

[REF-44] Michael Howard, David LeBlanc and John Viega. "24 Deadly Sins of Software Security". "Sin 2: Web-Server Related Vulnerabilities (XSS, XSRF, and Response Splitting)." Page 31. McGraw-Hill. 2010.

[REF-44] Michael Howard, David LeBlanc and John Viega. "24 Deadly Sins of Software Security". "Sin 3: Web-Client Related Vulnerabilities (XSS)." Page 63. McGraw-Hill. 2010.

OF.

[REF-712] "Cross-site scripting". Wikipedia. 2008-08-26. <[https://en.wikipedia.org/wiki/Cross-site\\_scripting](https://en.wikipedia.org/wiki/Cross-site_scripting)>. URL validated: 2023-04-07.

[REF-7] Michael Howard and David LeBlanc. "Writing Secure Code". Chapter 13, "Web-Specific Input Issues" Page 413. 2nd Edition. Microsoft Press. 2002-12-04. <<https://www.microsoftpressstore.com/store/writing-secure-code-9780735617223>>.

[REF-714] RSnake. "XSS (Cross Site Scripting) Cheat Sheet". <<http://ha.ckers.org/xss.html>>.

TEXT.

[REF-715] Microsoft. "Mitigating Cross-site Scripting With HTTP-only Cookies". <[https://learn.microsoft.com/en-us/previous-versions//ms533046\(v=vs.85\)?redirectedfrom=MSDN](https://learn.microsoft.com/en-us/previous-versions//ms533046(v=vs.85)?redirectedfrom=MSDN)>. URL validated: 2023-04-07.

[REF-716] Mark Curphey, Microsoft. "Anti-XSS 3.0 Beta and CAT.NET Community Technology Preview now Live!". <<https://learn.microsoft.com/en-us/archive/blogs/cisg/anti-xss-3-0-beta-and-cat-net-community-technology-preview-now-live>>. URL validated: 2023-04-07.

[REF-45] OWASP. "OWASP Enterprise Security API (ESAPI) Project". <<http://www.owasp.org/index.php/ESAPI>>.



# CWE-79, Part 6

Modifications –  
most recent should  
probably be on top

Not sure who  
uses or wants  
these details  
(besides me  
about once a  
month)

## ▼ Content History

### ▼ Submissions

Submission Date	Submitter	Organization
2006-07-19 (CWE Draft 3, 2006-07-19)	PLOVER	

### ▼ Modifications

Modification Date	Modifier	Organization
2008-07-01 (CWE 1.0, 2008-09-09)	Eric Dalci updated Time_of_Introduction	Cigital
2008-08-15 (CWE 1.0, 2008-09-09)	Suggested OWASP Top Ten 2004 mapping	Veracode
2008-09-08	CWE Content Team updated Alternate_Terms, Applicable_Platforms, Background_Details, Common_Consequences, Description, Relationships, Other_Notes, References, Taxonomy_Mappings, Weakness_Ordinalities	MITRE
2009-01-12	CWE Content Team updated Alternate_Terms, Applicable_Platforms, Background_Details, Common_Consequences, Demonstrative_Examples, Description, Detection_Factors, Enabling_Factors_for_Exploitation, Name, Observed_Examples, Other_Notes, Potential_Mitigations, References, Relationships	MITRE
2009-03-10	CWE Content Team updated Potential_Mitigations	MITRE
2009-05-27	CWE Content Team updated Name	MITRE
2009-07-27	CWE Content Team updated Description	MITRE
2009-10-29	CWE Content Team updated Observed_Examples, Relationships	MITRE
2009-12-28	CWE Content Team updated Demonstrative_Examples, Description, Detection_Factors, Enabling_Factors_for_Exploitation, Observed_Examples	MITRE



# Category Example

Numeric  
ID  
displayed

Repetitive  
“HasMem  
ber”

Yellow  
background  
is different,  
but  
structure /  
layout is  
the same

## CWE CATEGORY: Audit / Logging Errors

Category ID: 1210

Vulnerability Mapping: **PROHIBITED**

### ▼ Summary

Weaknesses in this category are related to audit-based components of a software system. Frequently these deal with logging user activities in order to identify undesired access and modifications to the system. The weaknesses in this category could lead to a degradation of the quality of the audit capability if they are not addressed.

### ▼ Membership

Nature	Type	ID	Name
MemberOf	V	699	<a href="#">Software Development</a>
HasMember	B	117	<a href="#">Improper Output Neutralization for Logs</a>
HasMember	B	222	<a href="#">Truncation of Security-relevant Information</a>
HasMember	B	223	<a href="#">Omission of Security-relevant Information</a>
HasMember	B	224	<a href="#">Obscured Security-relevant Information by Alternate Name</a>
HasMember	B	778	<a href="#">Insufficient Logging</a>
HasMember	B	779	<a href="#">Logging of Excessive Data</a>

So many links  
with purple text

### ▼ Vulnerability Mapping Notes

**Usage: PROHIBITED** (this CWE ID must not be used to map to real-world vulnerabilities)



# View-699 Example – Part 1

Numeric  
ID  
displayed

## CWE VIEW: Software Development

View ID: 699

Vulnerability Mapping: **PROHIBITED**

Type: Graph

Downloads: [Booklet](#) | [CSV](#) | [XML](#)

### ▼ Objective

This view organizes weaknesses around concepts that are frequently used or encountered in software development. This includes all aspects of the software development lifecycle including both architecture and implementation. Accordingly, this view can align closely with the perspectives of architects, developers, educators, and assessment vendors. It provides a variety of categories that are intended to simplify navigation, browsing, and mapping.

### ▼ Audience

Stakeholder	Description
Software Developers	Software developers (including architects, designers, coders, and testers) use this view to better understand potential mistakes that can be made in specific areas of their software application. The use of concepts that developers are familiar with makes it easier to navigate this view, and filtering by Modes of Introduction can enable focus on a specific phase of the development lifecycle.
Educators	Educators use this view to teach future developers about the types of mistakes that are commonly made within specific parts of a codebase.

### ▼ Relationships

The following graph shows the tree-like relationships between weaknesses that exist at different levels of abstraction. At the highest level, categories and pillars exist to group weaknesses. Categories (which are not technically weaknesses) are special CWE entries used to group weaknesses that share a common characteristic. Pillars are weaknesses that are described in the most abstract fashion. Below these top-level entries are weaknesses at varying levels of abstraction. Classes are still very abstract, typically independent of any specific language or technology. Base level weaknesses are used to present a more

Data  
downloads  
not clear

Long  
explanatory  
text





# View-699 Example – Part 2

language or technology. Base level weaknesses are used to present a more specific type of weakness. A variant is a weakness that is described at a very low level of detail, typically limited to a specific language or technology. A chain is a set of weaknesses that must be reachable consecutively in order to produce an exploitable vulnerability. While a composite is a set of weaknesses that must all be present simultaneously in order to produce an exploitable vulnerability.

Show Details: ☐

[Expand All](#) | [Collapse All](#) | [Filter View](#)

## 699 - Software Development

- ☐ ☒ API / Function Errors - (1228)
- ☐ ☒ Audit / Logging Errors - (1210)
- ☐ ☒ Authentication Errors - (1211)
- ☐ ☒ Authorization Errors - (1212)
- ☐ ☒ Bad Coding Practices - (1006)
- ☐ ☒ Behavioral Problems - (438)
- ☐ ☒ Business Logic Errors - (840)
- ☐ ☒ Communication Channel Errors - (417)
- ☐ ☒ Complexity Issues - (1226)
- ☐ ☒ Concurrency Issues - (557)
- ☐ ☒ Credentials Management Errors - (255)
- ☐ ☒ Cryptographic Issues - (310)
- ☐ ☒ Key Management Errors - (320)
- ☐ ☒ Data Integrity Issues - (1214)
- ☐ ☒ Data Processing Errors - (19)
- ☐ ☒ Data Neutralization Issues - (137)
- ☐ ☒ Documentation Issues - (1225)
- ☐ ☒ File Handling Issues - (1219)
- ☐ ☒ Encapsulation Issues - (1227)
- ☐ ☒ Error Conditions, Return Values, Status Codes - (389)
- ☐ ☒ Expression Issues - (569)

“Expand All” /  
other options  
kind of  
obscured

“Show  
Details” not  
clear



# Make Useful Features More Readily Apparent

- **There's lots of potentially helpful data that casual users won't be exposed to**
- **ID lookup (always in the top corner)**
- **Downloads of entire repository**
  - Diff files, older versions
- **Individual views**
  - Download data (CSV, XML, PDF, etc.) – small links in upper right
- **Expose REST API option**
- **Hierarchy visualizations (PDFs) for various views**
- **View filter**
  - Option is in a clickable link in the middle of the page
- **Google keyword search**
  - We don't (yet) have logs on what people search for
  - Could consider preventing indexing of some CWEs, or restricting results to only weaknesses



# Topic 2

## Open Discussion

*Chris Coffin*





# Next Meeting – March 26 @ 12pm

---

**PLEASE CONTACT WITH ANY QUESTIONS OR THOUGHTS**

**CWE@MITRE.ORG**



CWE is sponsored by [U.S. Department of Homeland Security](#) (DHS) [Cybersecurity and Infrastructure Security Agency](#) (CISA). Copyright © 1999–2024, [The MITRE Corporation](#). CWE and the CWE logo are trademarks of The MITRE Corporation.

# Backups



CWE is sponsored by [U.S. Department of Homeland Security](#) (DHS) [Cybersecurity and Infrastructure Security Agency](#) (CISA). Copyright © 1999–2024, [The MITRE Corporation](#). CWE and the CWE logo are trademarks of The MITRE Corporation.

# Reminder

---

- VulnCon 2025 is April 7-10, 2025
- <https://www.first.org/conference/vulncon2025/>
- Call for papers was extended to January 31st



# Home Page Redesign

---

- **The Problem: Lack of direction “where do I go now?” - especially for new users**
- **Proposed solution: Focus on grouping sections by common user actions**
  - Learn About CWE
  - Access Content
  - Contribute
    - Contribute content
    - Participate in working groups



# Reorganize New User Content

---

- **The Problem: New user content is in multiple places (under About and under Root Cause Mapping)**
- **Proposed Solution: Reorganize new user content**
  - Remove duplication (e.g., New to CWE and Root Cause Mapping Guidance)
  - Tie together pieces from different places
  - Specific information focused on different types of users



# New User Guidance on CWE Entry Pages

---

- **The Problem:** many new users enter the CWE website through a link to a specific CWE weakness page. No clear guidance without taking the time to read through all the new user info (which many won't have the time to do)
- **Proposed Solution: Add some content and structure to the CWE pages**
  - Make it easier for the novice to find what they need quickly
  - Could add some additional categorization to the CWE weakness entries as proposed on the following two slides



# CWE Element/Information Presentation – Mockup for New CWE Users

## CWE-125: Out-of-bounds Read

Weakness ID: 125

Abstraction: Base

Structure: Simple

View customized information:

Conceptual

Operational

Mapping Friendly

Complete

Custom

### What is the Weakness?

▼ Description

The product reads data past the end, or before the beginning, of the intended buffer.

▼ Extended Description

Typically, this can allow attackers to read sensitive information from other memory locations or cause a crash. A crash can occur when the code reads a variable amount of data and assumes that a sentinel exists to stop the read operation, such as a NUL in a string. The expected sentinel might not be located in the out-of-bounds memory, causing excessive data to be read, leading to a segmentation fault or a buffer overflow. The product may modify an index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent read operation then produces undefined or unexpected results.

### How can the Weakness affect me?

▼ Common Consequences

<div><div></div></div> Scope	Impact	Likelihood
Confidentiality	Technical Impact: Read Memory	
	Technical Impact: Bypass Protection Mechanism	
Confidentiality	By reading out-of-bounds memory, an attacker might be able to get secret values, such as memory addresses, which can be bypass protection mechanisms such as ASLR in order to improve the reliability and likelihood of exploiting a separate weakness to achieve code execution instead of just denial of service.	

▼ Demonstrative Examples

Example 1










# CWE Element/Information Presentation – Mockup for New CWE Users

index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent read operation then produces undefined or unexpected results.

### How does this Weakness relate to others?

#### ▼ Relationships

**Relevant to the view "Research Concepts" (CWE-1000)**

Nature	Type	ID	Name
ChildOf		119	<a href="#">Improper Restriction of Operations within the Bounds of a Memory Buffer</a>
ParentOf		126	<a href="#">Buffer Over-read</a>
ParentOf		127	<a href="#">Buffer Under-read</a>
CanFollow		822	<a href="#">Untrusted Pointer Dereference</a>
CanFollow		823	<a href="#">Use of Out-of-range Pointer Offset</a>
CanFollow		824	<a href="#">Access of Uninitialized Pointer</a>
CanFollow		825	<a href="#">Expired Pointer Dereference</a>

**Relevant to the view "Software Development" (CWE-699)**

Nature	Type	ID	Name
MemberOf		1218	<a href="#">Memory Buffer Errors</a>

**Relevant to the view "Weaknesses for Simplified Mapping of Published Vulnerabilities" (CWE-1003)**

**Relevant to the view "CISQ Quality Measures (2020)" (CWE-1305)**

**Relevant to the view "CISQ Data Protection Measures" (CWE-1340)**

### Where can I get more information?

#### ▼ References

[REF-1034] Raoul Strackx, Yves Younan, Pieter Philippaerts, Frank Piessens, Sven Lachmund and Thomas Walter. "Breaking the memory secrecy assumption". ACM. 2009-03-31. <<https://dl.acm.org/doi/10.1145/1519144.1519145>>. URL validated: 2023-04-07.

[REF-1035] Fermin J. Serna. "The info leak era on software exploitation". 2012-07-25. <[https://media.blackhat.com/bh-us-12/Briefings/Cerna/BH-US-12-Cerna\\_Leak\\_Era\\_Slides.pdf](https://media.blackhat.com/bh-us-12/Briefings/Cerna/BH-US-12-Cerna_Leak_Era_Slides.pdf)>.





# Improve Search

---

- **The Problem: Search is a generic option that searches the whole website, without the ability to focus in on key items of interest**
- **Proposed Solution: Build a more robust search capability**
  - Advanced search, filter results
    - Select fields to search in
    - Search alternate terms
    - Show only mappable CWEs



# What's new in CWE Release 4.15?

---

- **First installment of major usability improvements that are underway to enhance the understandability, navigability, and usability of CWE content**
- **15 CWE Entry pages now include a concise summary of the weakness along with a visual aid at the top of each entry page**
  - These 15 CWEs were selected based on their inclusion in the CWE top 25 list
  - Full list can be found at:  
[https://cwe.mitre.org/news/archives/news2024.html#july16\\_CWE\\_Version\\_4.15\\_Now\\_Available](https://cwe.mitre.org/news/archives/news2024.html#july16_CWE_Version_4.15_Now_Available)
- **The Alternate Terms, Common Consequences, and Mitigations sections were reordered so that they appear directly below the new summary section**



# Open Discussion Topics?

---

- **Increase motivations to fix weaknesses versus vulnerabilities**
  - Could we find a way to incorporate cost and expense associated with a weakness into CWE?
- **What CWE personas most benefit from weakness visualizations?**
  - Technical writers and security architects whose audience is less technical could benefit from reusing visualizations
  - New or junior level Software or Hardware developers could better understand security issues in visual form
- **CWE persona for technical manager?**



# Usability: Macro- and Micro-Level Review

---

- **Macro –**

- How to organize weaknesses at structural, site-wide level?
  - Organizing Views
- Site-wide navigation

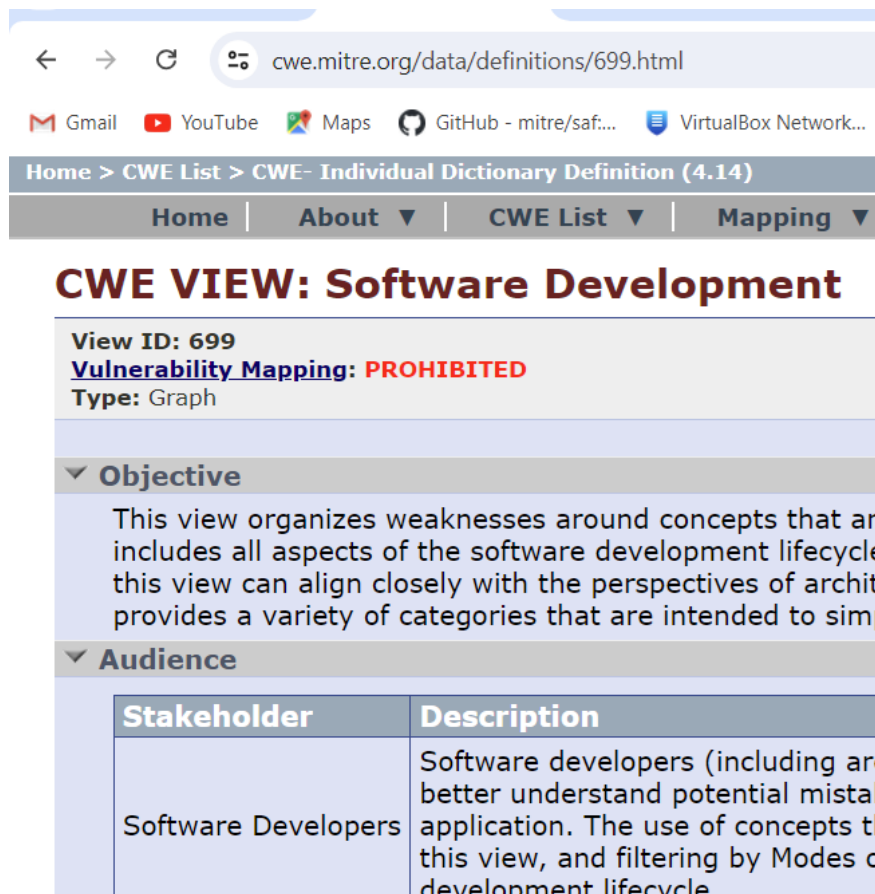
- **Micro –**

- How to define a weakness understandably/accurately
- Remove redundant and unnecessary information
  - Moving information to a better place
- Better leverage the schema



# View and Category IDs

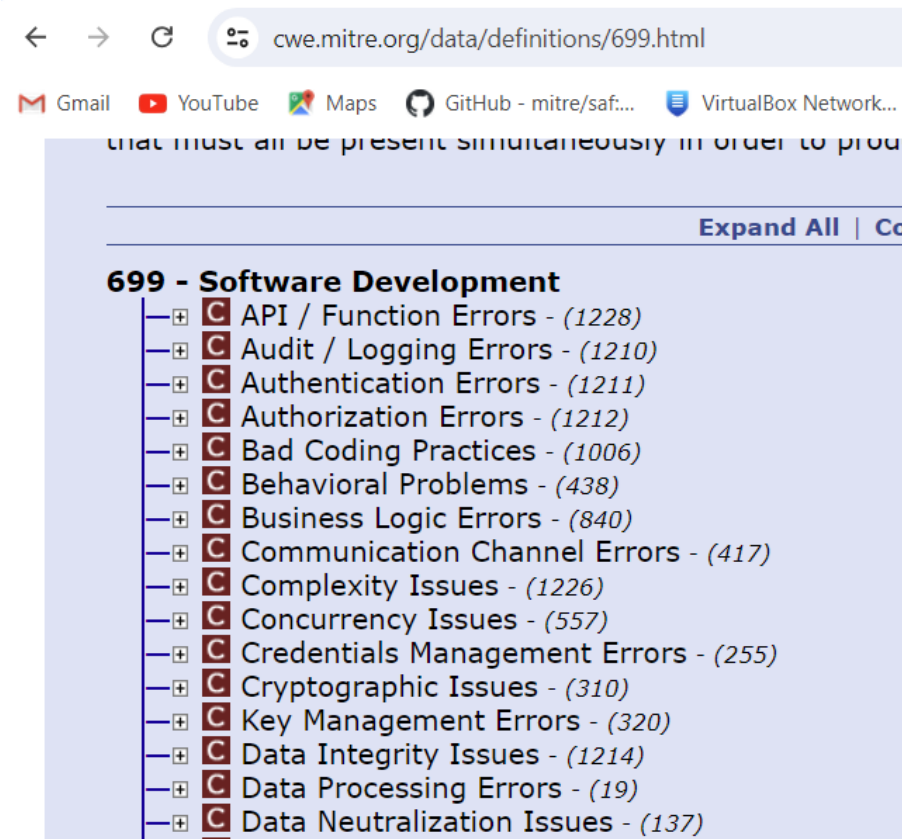
- Discussions around removing View and/or Category CWE IDs from page (under title) to prevent mappings
- Already removed IDs from titles
- IDs will still be used in URL



The screenshot shows the 'CWE VIEW: Software Development' page on cwe.mitre.org. The URL is cwe.mitre.org/data/definitions/699.html. The page has a navigation bar with links: Home, About, CWE List, and Mapping. The main content area is titled 'CWE VIEW: Software Development' and includes the following information:

- View ID: 699
- Vulnerability Mapping: PROHIBITED
- Type: Graph
- Objective: This view organizes weaknesses around concepts that are... includes all aspects of the software development lifecycle... this view can align closely with the perspectives of architecture... provides a variety of categories that are intended to simplify...
- Audience: A table with two columns: Stakeholder and Description.

Stakeholder	Description
Software Developers	Software developers (including architects) better understand potential mistakes in application. The use of concepts in this view, and filtering by Modes of development lifecycle.



The screenshot shows the '699 - Software Development' list on cwe.mitre.org. The URL is cwe.mitre.org/data/definitions/699.html. The list includes the following items:

- API / Function Errors - (1228)
- Audit / Logging Errors - (1210)
- Authentication Errors - (1211)
- Authorization Errors - (1212)
- Bad Coding Practices - (1006)
- Behavioral Problems - (438)
- Business Logic Errors - (840)
- Communication Channel Errors - (417)
- Complexity Issues - (1226)
- Concurrency Issues - (557)
- Credentials Management Errors - (255)
- Cryptographic Issues - (310)
- Key Management Errors - (320)
- Data Integrity Issues - (1214)
- Data Processing Errors - (19)
- Data Neutralization Issues - (137)



# Other Macro Changes?

---

- Any other thoughts or ideas on Macro usability changes?



# UEWG: Purpose

---

- **Mission:** Identifying areas where CWE content, rules, guidelines, and best practices must improve to better support stakeholder community, and work collaboratively to fix them
- **Periodic reporting of activities to CWE Board**
  - Next quarterly Board meeting TBD Q2-2024
- **Please solicit participation from your contacts**
  - Contact: [cwe@mitre.org](mailto:cwe@mitre.org)



# Housekeeping

---

- **CWE UEWG May Meeting**
  - The next regularly scheduled meeting is Wed 7/31
- **The CWE Program is continuously seeking feedback on UEWG activities and priorities during these sessions or via email: [cwe@mitre.org](mailto:cwe@mitre.org)**





# Usability Task Micro Updates Initial Goals

## Above the fold (before the webpage scroll point):

- Important and concise text is above the fold so the reader can easily scan and digest

## Points to Make Above the Fold:

- **Describe just the weakness and provide a visual aid**
  - Concise summary of weakness (only in description, no extended description)
  - Describe the condition and some context about the condition
  - Concise is 3 – 4 sentences. Images will provide additional context.

## Reorder Elements:

- **Alternate Terms**
- **Consequences Element** (bad outcomes)
- **Mitigations Element** (what to do about the weakness)
- Remaining Elements follow



# Prioritizing User Experience Improvements

- **User Experience Working Group was established in response to negative feedback related to CWE's usability**
  - This group has accomplished much in response (e.g., submission guidelines, mapping guidance, "New to CWE?" launch, weakness filtering)
  - We must now begin to address usability and understandability in further ways throughout the corpus structure and down to individual weakness language
- **CWE entry clean-up (across the entire corpus)**
  - Understandability: revising all entries for simpler language; remove redundancy
  - Simplifying descriptions/extended descriptions to leverage other elements appropriately
    - E.g., removing example instance, impacts, etc. language from descriptions
    - Only having one description written in concise, accurate language focusing on the weakness
  - Completeness: ensuring all entries have basic required elements populated
  - Visualizations: adding to entries to explain topics visually (leverage Abhi's top25 examples, build from there)



# Usability: Macro- and Micro-Level

---

- **Macro –**

- How to organize weaknesses at structural, site-wide level?
  - Organizing Views
- Site-wide navigation

- **Micro –**

- How to define a weakness understandably/accurately
- Remove redundant and unnecessary information
- Better leverage the schema



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

Weakness ID: 89

Abstraction: Base  
Structure: Simple

## Description

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

## Extended Description

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

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

## Relationships



### Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	🟢	943	<a href="#">Improper Neutralization of Special Elements in Data Query Logic</a>
ParentOf	🟡	564	<a href="#">SQL Injection: Hibernate</a>
CanFollow	🟡	456	<a href="#">Missing Initialization of a Variable</a>



CWE is sponsored by [U.S. Department of Homeland Security](#) (2024, [The MITRE Corporation](#)). CWE and the CWE logo are tra

# CWE-89: Improper Neutralization of Special Elements used in an SQL Command

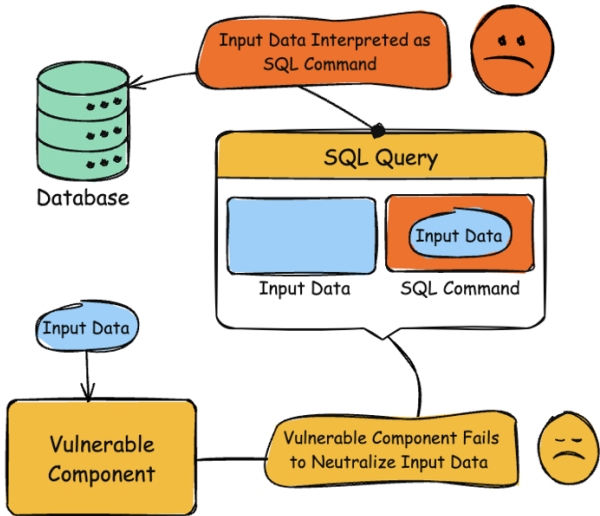
Weakness ID: 89

Usage for Mapping: **ALLOWED** (this CWE ID may be used to map to real-world vulnerabilities)

Abstraction: Base

## Description

The product uses user-controllable input to construct an SQL command incorrectly or without neutralizing its special elements. This can cause inputs to be interpreted as SQL code instead of ordinary user data. This weakness depends on the fact that SQL makes no real distinction between the control and data planes.



## Alternate Terms

'SQL Injection'

## Common Consequences



Scope	Impact
Confidentiality	<b>Technical Impact:</b> Read Application Data A user could construct an SQL command to reveal sensitive information stored in a database

# Overall Prioritization

---

- At what level should we begin our focus?
- Macro (CWE organization and navigation) vs Micro level (weakness understandability)
- Micro could include CWE record:
  - Understandability
  - Completeness
  - Visualizations



# A possible path forward

---

- **Discuss a single CWE and how we could improve usability by**
  - Merging the description and extended descriptions
  - Removing redundant information
  - Moving information that belongs in other parts of the schema
  - Removing vulnerability specific information
- **We could walk through one example in the meeting**
- **Would anyone be interested in trying this exercise with another CWE on their own**
  - Could present their results in the next meeting



# Topic 2

## New to CWE Updates

*Chris Coffin*



# Prioritizing User Experience Improvements

---

- **To come: New Strategy and Implementation Plan for usability improvements across CWE corpus aligned to required elements and their requirements as outlined in CWE content suggestion guidelines**
- **Community partners in UEWG are invited to help us tackle this important work**
  - More intentional focus in UEWG
  - CWE Content Web Submission Form for suggested mods on any weakness
  - Content Development Repository (CDR), GitHub repo for community collaboration is coming soon and will provide transparent, collaborative platform on which to work





# Recent Discussions on SW Licensing Issues

- **Since November, a resurrection of debate around SW licensing issues being in/out of CWE scope on the CWE-Research email listserv**
- **In 2018, it was determined that “improper licensing” was outside of CWE's scope**
  - Rationale: Impact to software and its usage not through the technical exploitation of a software security weakness in architecture, design, or code. Rather, it is through policy/programmatic exploitation.
  - Availability concern comparable with supply chain issues where one disrupts a supplier to stop/limit a product from being delivered, and hence make it not available
- **Recent arguments are varied:**
  - Pro: Misusing SW licenses can negatively affect maintainability, availability
  - Against: Invalid/Improper licenses cannot lead to a vulnerability
  - Pro: CWE absorbed CQE content ~2018; improper licensing is a code quality issue
  - Against: Licensing issues are not a property of SW, but of the society and economy around the SW



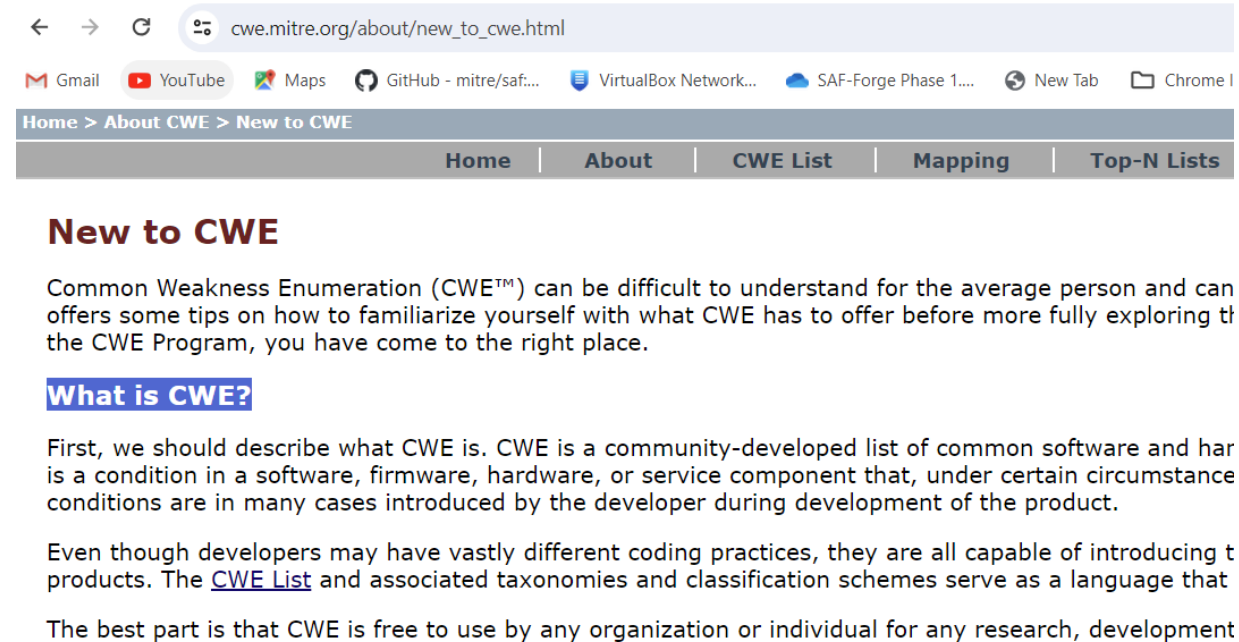
# CWE Scope

- **CWE's definition of "Weakness":**
  - A condition in a software, firmware, hardware, or service component that, under certain circumstances, could contribute to the introduction of vulnerabilities
- **Community members sometimes suggest new entries to CWE that do not satisfy this "weakness" definition, but they want CWE to treat them as "weaknesses"**
- **Other times, people effectively suggest the expansion of CWE's scope beyond "traditional" software/hardware**
- **"Scope exclusions" attempt to formalize decisions about what can or cannot be included in CWE as an official "weakness" entry**
  - Have already been used in external submissions



# New to CWE – Additional Examples

- The current New to CWE page has one example “CWE-798: Use of Hard-coded Credentials”
- More examples are needed
- Should be simple to understand to closely match the intended audience of the New to CWE page
- Examples for consideration?
  - Might focus on 2023 Top 25 CWEs?
  - CWE-434: Unrestricted Upload of File with Dangerous Type
  - CWE-287: Improper Authentication
  - Others?



# New to CWE – Additional Examples – Selection Criteria (Continued)

- What CWE characteristics are important when determining the best examples for new CWE users?
- Is the description easy to understand for a non-developer or industry expert?
- Does it include good examples (demonstrative and observed)?
- Is it a well-known/well-understood weakness?
- Does it exist in the Top 25 list (now or previous years)?
- Others?

## CWE-787: Out-of-bounds Write

Weakness ID: 787

Abstraction: Base

Structure: Simple

View customized information:

Conceptual

Operational

Mapping  
Friendly

Complete

Custom

### Description

The product writes data past the end, or before the beginning, of the intended buffer.

### Extended Description

Typically, this can result in corruption of data, a crash, or code execution. The product may modify an index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent write operation then produces undefined or unexpected results.

### Alternate Terms

**Memory Corruption:** Often used to describe the consequences of writing to memory outside the bounds of a buffer, or to memory that is invalid, when the root cause is something other than a sequential copy of excessive data from a fixed starting location. This may include issues such as incorrect pointer arithmetic, accessing invalid pointers due to incomplete initialization or memory release, etc.

### Relationships

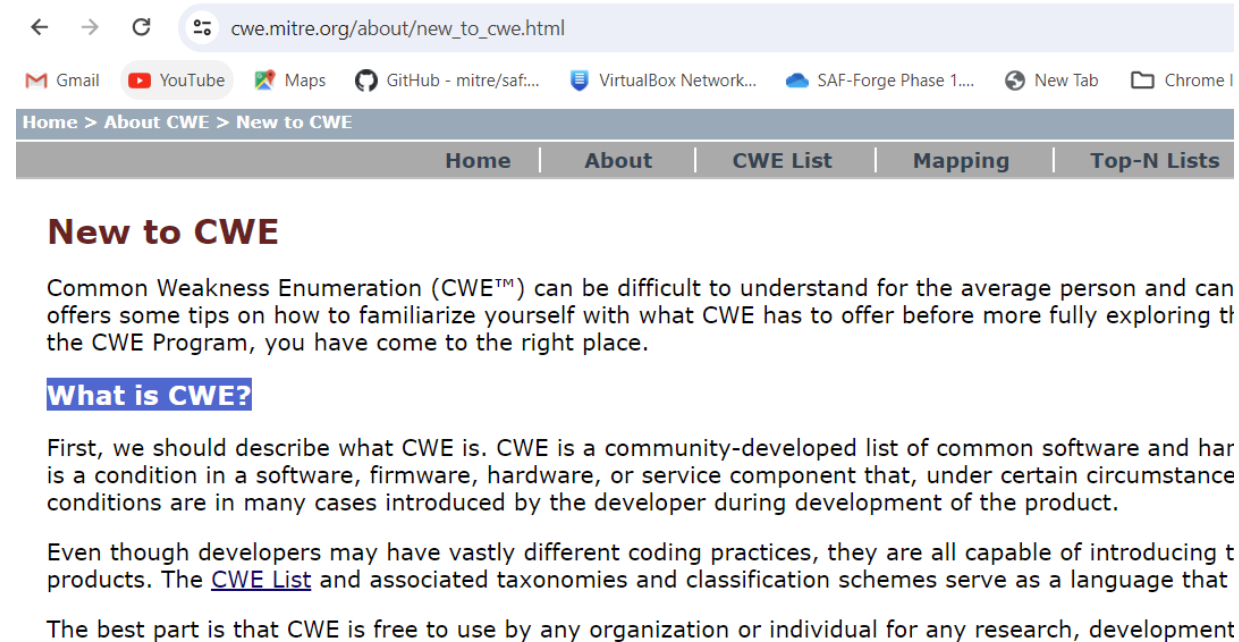
#### Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf	✓	119	<a href="#">Improper Restriction of Operations within the Bounds of a Memory Buffer</a>
ParentOf	✓	121	<a href="#">Stack-based Buffer Overflow</a>
ParentOf	✓	122	<a href="#">Heap-based Buffer Overflow</a>
ParentOf	ⓘ	123	<a href="#">Write-what-where Condition</a>
ParentOf	ⓘ	124	<a href="#">Buffer Underwrite ('Buffer Underflow')</a>
CanFollow	ⓘ	822	<a href="#">Untrusted Pointer Dereference</a>
CanFollow	ⓘ	823	<a href="#">Use of Out-of-range Pointer Offset</a>
CanFollow	ⓘ	824	<a href="#">Access of Uninitialized Pointer</a>



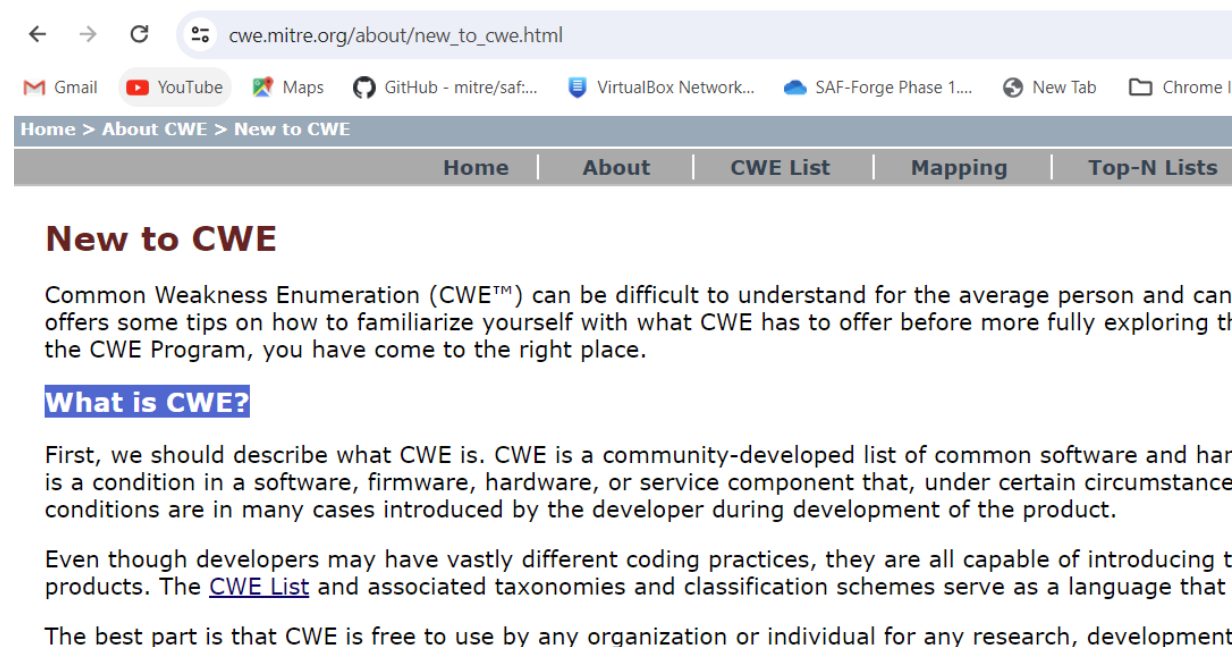
# New to CWE Series – Future Topics

- The CWE team is considering developing additional documents as part of a “New to CWE” series
- Keep them short so that new and casual users can read quickly
- Each document would link to others in the series where appropriate
- Topics for consideration?
  - Categorization - Views and Categories
  - How do I navigate the CWE corpus?
  - CWE Hierarchy - Pillars, Classes, Bases, and Variants
  - Others?



# New to CWE Series – How to Navigate CWE

- The CVE -> CWE Mapping Guidance exists on the website today
- The Mapping Methodologies section of the guidance could be reused with some modification
  - Keyword search
  - Views
  - PDF Visualization
- The goal is to create a “New to CWE: How to Navigate CWE” document that could be referenced by the current “New to CWE” page that exists today





# CWE Element/Information Presentation – Mockup for New CWE Users

## CWE-125: Out-of-bounds Read

**Weakness ID: 125**  
**Abstraction:** Base  
**Structure:** Simple

View customized information:

Conceptual

Operational

Mapping Friendly

Complete

Custom

### What is the Weakness?

#### ▼ Description


The product reads data past the end, or before the beginning, of the intended buffer.

#### ▼ Extended Description

Typically, this can allow attackers to read sensitive information from other memory locations or cause a crash. A crash can occur when the code reads a variable amount of data and assumes that a sentinel exists to stop the read operation, such as a NUL in a string. The expected sentinel might not be located in the out-of-bounds memory, causing excessive data to be read, leading to a segmentation fault or a buffer overflow. The product may modify an index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent read operation then produces undefined or unexpected results.

### How can the Weakness affect me?

#### ▼ Common Consequences

 Scope	Impact	Likelihood
Confidentiality	<b>Technical Impact:</b> <i>Read Memory</i>	
	<b>Technical Impact:</b> <i>Bypass Protection Mechanism</i>	
Confidentiality	By reading out-of-bounds memory, an attacker might be able to get secret values, such as memory addresses, which can be bypass protection mechanisms such as ASLR in order to improve the reliability and likelihood of exploiting a separate weakness to achieve code execution instead of just denial of service.	

#### ▼ Demonstrative Examples

Example 1










# CWE Element/Information Presentation – Mockup for New CWE Users

index or perform pointer arithmetic that references a memory location that is outside of the boundaries of the buffer. A subsequent read operation then produces undefined or unexpected results.

## How does this Weakness relate to others?

### Relationships

#### Relevant to the view "Research Concepts" (CWE-1000)

Nature	Type	ID	Name
ChildOf		119	<a href="#">Improper Restriction of Operations within the Bounds of a Memory Buffer</a>
ParentOf		126	<a href="#">Buffer Over-read</a>
ParentOf		127	<a href="#">Buffer Under-read</a>
CanFollow		822	<a href="#">Untrusted Pointer Dereference</a>
CanFollow		823	<a href="#">Use of Out-of-range Pointer Offset</a>
CanFollow		824	<a href="#">Access of Uninitialized Pointer</a>
CanFollow		825	<a href="#">Expired Pointer Dereference</a>

#### Relevant to the view "Software Development" (CWE-699)

Nature	Type	ID	Name
MemberOf		1218	<a href="#">Memory Buffer Errors</a>

#### Relevant to the view "Weaknesses for Simplified Mapping of Published Vulnerabilities" (CWE-1003)

#### Relevant to the view "CISQ Quality Measures (2020)" (CWE-1305)

#### Relevant to the view "CISQ Data Protection Measures" (CWE-1340)

## Where can I get more information?

### References

[REF-1034] Raoul Strackx, Yves Younan, Pieter Philippaerts, Frank Piessens, Sven Lachmund and Thomas Walter. "Breaking the memory secrecy assumption". ACM. 2009-03-31. <<https://dl.acm.org/doi/10.1145/1519144.1519145>>. URL validated: 2023-04-07.

[REF-1035] Fermin J. Serna. "The info leak era on software exploitation". 2012-07-25. <[https://media.blackhat.com/bh-us-12/Briefings/Security/BU\\_US\\_12\\_Serna\\_Leak\\_Era\\_Slides.pdf](https://media.blackhat.com/bh-us-12/Briefings/Security/BU_US_12_Serna_Leak_Era_Slides.pdf)>.





# Grouping CWEs

CWE™ is a community-developed list of software and hardware weakness types. It serves as a common language, a

- **CWE entries are currently “grouped” in different ways to provide useful subsets of the CWE corpus for different purposes:**
  - Views: a subset of CWE entries that provides a way of examining CWE content. The two main view structures are Slices (flat lists) and Graphs (containing relationships between entries), examples include:
    - [CWE-1194: Hardware Design](#)
    - [CWE-699: Software Development](#)
    - [CWE-1400: Comprehensive Categorization for Software Assurance Trends](#)
    - [CWE-1003: Weaknesses for Simplified Mapping of Published Vulnerabilities](#) (NVD)
  - Categories: a CWE entry that contains a set of other entries that share a common characteristic. A category is not a weakness, but rather a structural item that helps users find weaknesses that share the stated common characteristic.
    - [CWE-1199: General Circuit and Logic Design Concerns](#)
  - ~ Overall Hierarchy
    - [CWE-1000: Research Concepts](#) contains all CWE entries in one hierarchical structure

# Grouping CWEs, cont.

---

- **What groupings are most useful to new or casual CWE users? Experienced users?**
- **How can groupings be better presented/discovered/identified to the user?**
- **Should new users be guided to groupings of CWEs for learning about CWE? (e.g., links in user stories)**
- **Are there additional groupings that we are missing? Too many?**



# CSV Single Colon Separators Within Column Data

- A double colon is used to separate csv fields/columns data, while a single colon is used to separate multi-value data within fields/columns
- The Observed Examples field contains Reference and link data that includes a url in some cases (colon within “http://...”)
- Should a note be added to the download data that warns the user of this, or should we look into an alternative separator?
- Example taken from CWE - CWE-41: Improper Resolution of Path Equivalence (4.12) (mitre.org)
  - ::REFERENCE:CVE-2000-1114:DESCRIPTION:Source code disclosure using trailing dot:LINK:https://www.cve.org/CVERecord?id=CVE-2000-1114::REFERENCE:CVE-2002-1986:DESCRIPTION:Source code disclosure using trailing



# CWE User Pain Points

---

- Pain point topics that the group is aware of or would like to discuss
- For those on the call, what were your biggest questions or concerns when beginning to use CWE?
- Are there common questions that CWE users have that are not covered in the current FAQ?
- Other potential opportunities:
  - Features we could expand or improve to make CWE consumption easier?
  - Maybe engage the community in one or more ways to solicit this kind of feedback (see topic #3)
- Other thoughts?



# Community Engagement Strategy

---

- **Develop a strategy for engaging the CWE user community for feedback**
- **What are the best methods to query the community on topics such as the pain points covered in topic #2**
- **What communication methods should be employed?**
  - E.g., polls, emails, web, social media
- **Should we target specific user types?**
  
- **Other thoughts?**



# CWE Video Tips Series

---

## ▪ Current video ideas:

- How to search CWE for a weakness
- How to display only the information that you need with presentation filters
- What is a weakness (vs a vulnerability)
- How are weaknesses organized
- What is a category (how is it different than a pillar)
- What are views
- How and why to use the research view
- Use cases for CWE (could user stories be used?)
- How do I submit an idea for a new weakness
- How can I improve the quality of existing weaknesses



# New to CWE – Future Content

---

- **The New to CWE content audience is different from what has been catered to previously**
- **The audience is the casual or new user to CWE or even the manager who makes security funding decisions**
- **The team has previously drafted material for the New to CWE audience that covers the CWE hierarchy**
  - Not yet released material
  - Do members agree that this topic should be covered for New to CWE?
- **Are there other topics that UEWG members feel strongly about or believe should be covered given the intended audience?**
- **Should there be a close coupling of the topics covered here with the CWE Video Tips series?**



# CWE Naming and Vulnerability Mapping

---

- **Being thinking about solutions for common and well-known issues surrounding use of CWE names and how to more easily map vulnerabilities to CWEs**
- **Current CWE structure is difficult to understand and use**
- **Community needs better root cause information for vulnerabilities**
- **Does CWE naming need a change or update to support easier mapping?**
  - Remove CWE names for Views and/or Categories?
  - New naming that embeds a structure (e.g., CWE-1234-1)

