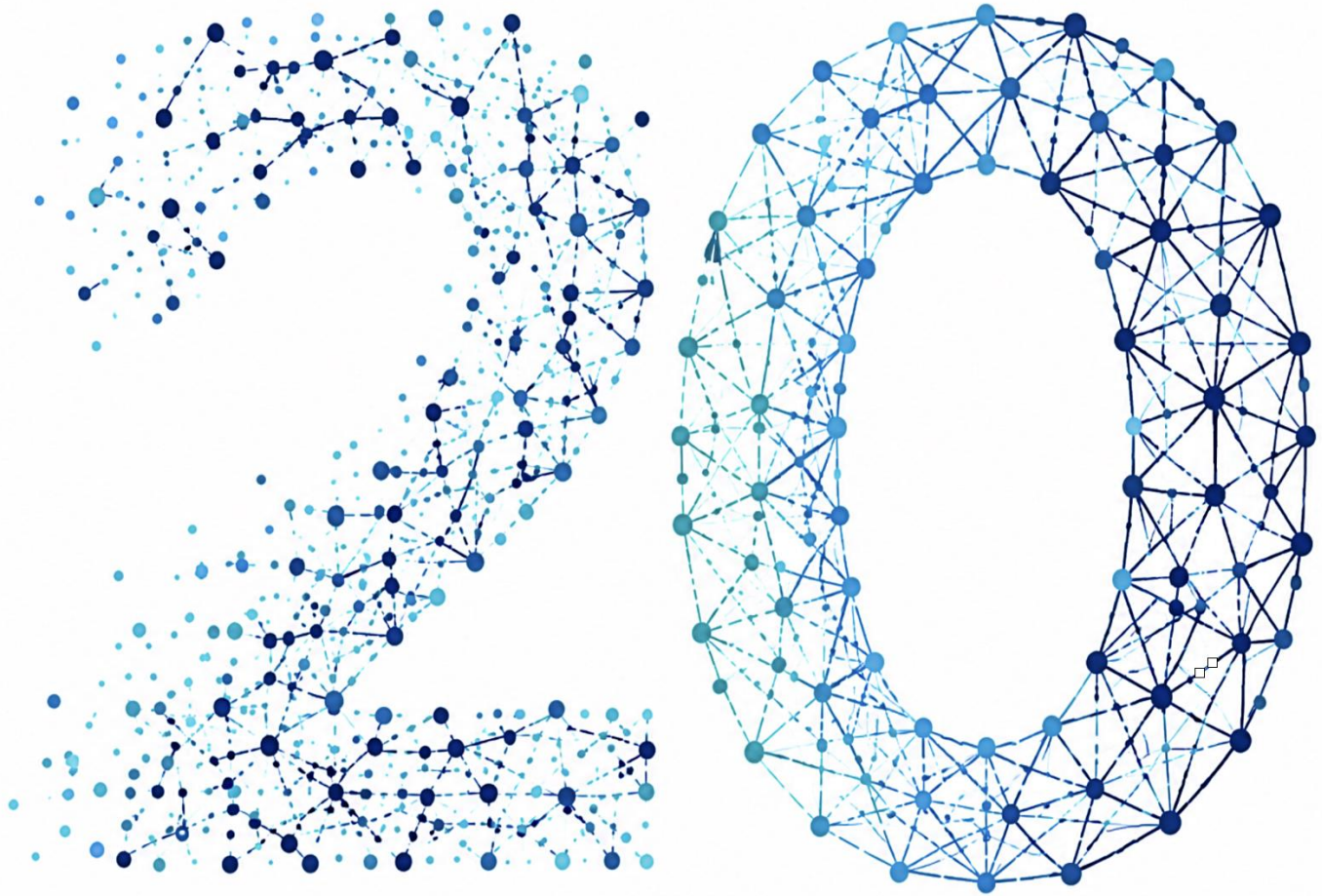




————— **YEARS** —————

2006 – 2026

TWENTY YEARS OF COMMUNITY-DEVELOPED
CYBERSECURITY WEAKNESS KNOWLEDGE

TABLE OF CONTENTS

3 — TWENTY YEARS OF CWE™

The idea behind a common language for understanding the weaknesses that lead to vulnerabilities

4 — BEFORE CWE HAD A NAME

PLOVER, early vulnerability research, and the accumulation of weakness knowledge

5 — 2006: CWE BECOMES PUBLIC

The first public drafts and the beginnings of a community-developed knowledge base

6 — ORGANIZING A COMMON LANGUAGE

How definitions, relationships, views, and supporting technical information made CWE useful

7 — FROM PRIORITIES TO PATTERNS

How weakness prioritization evolved from expert judgment to data-driven and context-specific analysis

8 — CWE REACHES HARDWARE

How community expertise expanded CWE beyond software and influenced the broader model

9 — TWENTY YEARS OF EVOLUTION

How the knowledge base evolved and CWE's spread across tools, standards, and security practice

10 — CWE AT 20

Twenty years of growth in the corpus, analytical use, digital reach, and machine consumption

11 — FROM KNOWLEDGE TO PRACTICE

How CWE moves into education, secure-development guidance, standards, and software assurance

12 — FROM ROOT CAUSE TO PREVENTION

Using recurring weakness information to feed product development and support Secure by Design

13 — MAKING CWE EASIER TO APPLY

Better mapping guidance, machine access, community contribution, and systematic corpus improvement

14 — BUILT WITH THE COMMUNITY

How the Board, Working Groups, SIGs, and contributors bring specialized expertise into CWE

15 — AI CHANGES THE LANDSCAPE, TWICE

AI/ML as a security domain—and AI as a new way to access and apply weakness knowledge

16 — CWE 5.0: BUILDING FOR THE NEXT ERA

Clearer concepts, richer relationships, broader lifecycle models, and more structured information

17 — THE NEXT TWENTY YEARS

Applying weakness knowledge in an era of AI-assisted development and increasing software scale





Twenty Years of CWE

In 2006, the cybersecurity community began building a shareable common language to describe the recurring mistakes and conditions that make software vulnerable.

Twenty years later, the Common Weakness Enumeration (CWE™) has evolved into a community-built knowledge base used across vulnerability root cause analysis, secure software and hardware development, security research, tooling, and education. CWE connects security problems found in various products with the broader weakness patterns that make those problems possible.

Vulnerability information tells us where a specific security problem has appeared. Weakness information helps explain what allowed it to happen. Examined across many vulnerabilities, consistent weakness information can reveal patterns that inform architecture, development, testing, technology choices, and other decisions made before similar vulnerabilities are repeated.

In 2026, the environment around CWE has changed considerably. Modern software development and deployment operate at unprecedented scale and speed, while stakeholder expectations for product security are much higher and more widespread. Hardware security has become a larger part of the cybersecurity landscape, while artificial intelligence is changing both the systems being secured and the tools available to secure them. At the same time, the growing volume of vulnerability information increasingly demands automation and new analytical methods.

Yet recurring weaknesses remain, and they contribute to the proliferation of vulnerabilities throughout the digital ecosystem. CWE gives the community a way to recognize those patterns, communicate about them, and build knowledge that can support remediation and prevention.

WHY IT MATTERS

CWE's original goals included providing a common language for cybersecurity weaknesses, a baseline for evaluating security tools, and support for identification, mitigation, and prevention.

Language changes the mental model.

The way a security problem is described influences where attention goes – to the adversary, the vulnerable product, the underlying weakness, or the opportunity to prevent recurrence.

ATTACK	VULNERABILITY	REMEDIATION	WEAKNESS(ES)	PREVENTION
What does an adversary do to <u>exploit</u> the condition?	Where, and how serious is the exploitable condition in a particular product or system?	What can be fixed or mitigated to address that specific vulnerable condition?	What underlying condition(s) <u>made it possible</u> ?	What can change so the same kind of weakness does not keep recurring?

-----CAPEC™-----CVE™-----CWE-----

Before CWE Had a Name

CWE grew out of earlier efforts to understand what vulnerability data could reveal about the causes of software security problems. Common Vulnerabilities and Exposures (CVE) had already demonstrated the value of a common identifier for individual publicly disclosed vulnerabilities. Yet researchers, developers, security-tool providers, and others lacked a sufficiently consistent way to describe the types of mistakes behind those vulnerabilities.

Software-security terminology varied widely. Similar problems could have different names, while familiar terms sometimes carried multiple meanings. Some descriptions focused on the method of attack or the consequence of exploitation rather than the condition that made exploitation possible. These differences complicated classification, tool comparison, communication, and efforts to carry lessons from vulnerability discovery back into development.

One important precursor was the Preliminary List of Vulnerability Examples for Researchers, or PLOVER. PLOVER began with observed security problems: more than 1,500 real-world CVE Records organized around roughly 290 types of weaknesses, faults, flaws, and related concepts.

The approach worked from individual examples toward recurring patterns. Many vulnerabilities may be distinct as product-level events while tracing back to similar mistakes. Once those mistakes can be described consistently, they can be studied across products and organizations and incorporated into security tools, development guidance, research, and training.

PLOVER and other foundational efforts – including Seven Pernicious Kingdoms, Software Fault Patterns, and NIST's SAMATE work – provided important source material, alongside early contributions from code analysis tool vendors. CWE built on that knowledge by giving weakness concepts clearer definitions, relationships, and structure.



Preliminary List of Vulnerability Examples for Researchers (PLOVER)

PLOVER - Preliminary List Of Vulnerability Examples for Researchers

[*] Author: Steve Christey (coley@mitre.org)
[*] Date: March 15, 2006
[*] Document Version: 0.24

Disclaimer: This is a DRAFT document that does not represent an official position of The MITRE Corporation. This document has been created to spur progress in vulnerability classification and vulnerability research.

SECTION 1. [INTRO] Introduction

Recently, there has been renewed interest in the classification and categorization of vulnerabilities, attacks, faults, and other concepts. Past efforts have largely focused on high-level theories, taxonomies, or schemes that do not sufficiently cover the wide variety of security issues that are found in today's products.

PLOVER - the Preliminary List Of Vulnerability Examples for Researchers - is a working document that lists over 1400 diverse, real-world examples of vulnerabilities, identified by their CVE number. The vulnerabilities are organized within a novel, detailed conceptual framework. The framework does not solve the entire classification problem, but it provides useful discussion points and an effective vocabulary for describing vulnerabilities at a low level of detail.

March 2006

FROM INSTANCES TO PATTERNS

1,500+

observed vulnerability examples



~290

recurring PLOVER types



Additional data sources and community work



2006
CWE

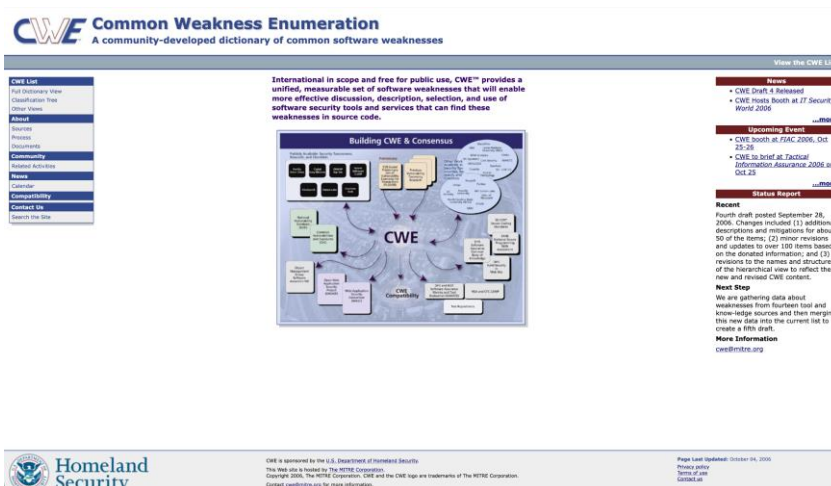
CWE launches as a public effort

CWE Becomes Public

The first public CWE draft appeared on the CVE website on March 15, 2006. Additional drafts followed as the emerging corpus incorporated existing taxonomies, security-tool knowledge, research, and community feedback.

On September 22, 2006, the dedicated CWE website launched. By the end of the year, CWE had established a public identity and a growing body of content.

"A community-developed dictionary of common software weaknesses."



An early version of the dedicated CWE website, October 2006.

MARCH 15, 2006	First public CWE draft
SPRING-SUMMER	Successive drafts and community input
SEPTEMBER 22	Dedicated website launches
LATE 2006	Continued corpus and compatibility work

"From a technical perspective, CWE has evolved through several stages: from early classification research and making sense of publicly reported vulnerabilities, to supporting developers and making weakness knowledge accessible to a much broader community."

Steve Christey Coley
CWE Co-Founder / Technical Lead

"DHS originally sponsored CWE to help address vulnerabilities at their root cause. CVE would identify specific vulnerability instances; CWE would identify the underlying weaknesses that cause them. Today, both remain critical: CVE supports tracking and remediation, while CWE helps developers and tools reduce recurring design and coding weaknesses before products ship."

Joe Jarzombek
Founding DHS Sponsor
Representative for CWE,
CWE Board Member



Organizing a Common Language

As CWE grew, users needed different ways to navigate and apply the same body of weakness knowledge. A developer may want to know where a weakness enters the development lifecycle and how it can be avoided. A vulnerability analyst needs enough precision to identify an appropriate root cause. Architects, hardware engineers, researchers, educators, and tool developers bring other questions.

CWE Version 1.0, released in September 2008, marked an important stage in that evolution. Its Development Concepts and Research Concepts views offered different ways to organize the corpus, while schema and relationship improvements gave the content greater structure.

The content of individual entries also became richer over time. Definitions could be supported by related weaknesses, consequences, modes of introduction, examples, detection methods, mitigations, mapping guidance, platform information, and references.

The structure beneath CWE evolved along with its content. Early schema changes gave the Program better ways to distinguish and connect weaknesses, views, categories, and other concepts. Later generations added greater depth and specificity to some elements, in addition to structure for new needs such as vulnerability-mapping guidance and richer content history, while preserving CWE as a consistent, machine-readable knowledge base.

One knowledge base, many ways to use it

DEVELOPER • Examples • Mitigations • Modes of Introduction	ARCHITECT • Consequences • Lifecycle Context	VULNERABILITY ANALYST / PSIRT • Definition • Abstraction • Mapping Guidance
TOOL PROVIDER • Identifiers • Structured Data	CWE ENTRY Definitions + Relationships + Technical Context	HARDWARE ENGINEER - Domain Context • Detection • Relationships
RESEARCHER • Views • Taxonomy • References	EDUCATOR • Examples • Concepts • Common Terminology	COMMUNITY CONTRIBUTOR • Content Development Repository (CDR) • Technical Review • Proposals

LOOKING AHEAD

As CWE's user community and use cases continue to grow, so do the demands placed on the knowledge base. CWE 5.0 is designed to improve content coverage and quality while better supporting different technologies, development lifecycles, and the needs of both human and automated consumption.



From Priorities to Patterns

The first CWE Top 25, published in collaboration with SANS in 2009, used software-security expertise to focus attention on programming errors considered widespread, consequential, and preventable. In 2019, its methodology changed. Published vulnerability information, CWE mappings, and severity data created a repeatable, data-driven ranking based on weakness patterns represented in real-world vulnerabilities.

Later editions placed greater emphasis on mapping quality and on what the underlying data could reveal beyond the ranking itself. The 2025 Top 25 analyzed root-cause mappings from 39,080 CVE Records while preserving more of the specificity in reported mappings rather than automatically rolling them into a smaller set of higher-level CWEs. That greater fidelity helps reveal persistent weakness patterns, changes over time, and places where vulnerability information may still be insufficient for confident root-cause analysis.

BEYOND THE RANKING

The Top 25 is one form of weakness prioritization. CWSS and CWRAP explored a more tailored approach, combining weakness scoring with business, mission, and technology context. Developed in an earlier phase of CWE, they remain useful examples of how weakness priorities can vary by environment rather than being captured by a single universal ranking.

ROOT CAUSE MAPPING

Accurate mappings connect individual vulnerabilities to underlying weakness concepts, creating a basis for pattern analysis and prevention.

39,080 CVE RECORDS made up the 2025 Top 25 –
An empirical base for weakness prioritization and trend analysis.

How the Top 25 methodology evolved

- **2009** Expert-informed, survey-based prioritization
- **2019** New, data-driven methodology
- **2020** Greater mapping fidelity and trend analysis
- **2026** Reimagining weakness prioritization with exploit information

“

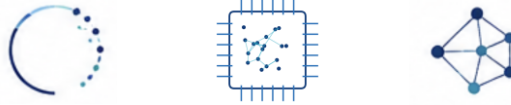
Root-cause information turns a single fix into organizational learning. Identifying recurring CWE "hot spots" across teams and codebases creates opportunities for targeted developer education or broader security engineering such as safer frameworks or guardrails. Understanding root-cause is what lets a security program measure whether it's actually getting better over time, versus just counting closed tickets.

Chris Eng

CWE Board Member



CWE Reaches Hardware



In February 2020, community collaboration helped expand CWE beyond its original software focus, with CWE 4.0 introducing hardware design weaknesses into the knowledge base.

Hardware brought different engineering practices, lifecycle stages, and security concerns. Existing software concepts did not automatically capture everything hardware designers and verification engineers needed to describe.

The Hardware CWE Special Interest Group (SIG) brought together expertise from industry, government, academia, and research organizations to develop content and improve how hardware weaknesses are represented in CWE.

Hardware CWE information has also begun to move into EDA and pre-silicon security verification workflows, where some commercial tools use CWE to define security requirements, map automated checks, and report coverage against known hardware weakness types.

The Most Important Hardware Weaknesses (MIHW) effort illustrates the balance required in this domain. Combining expert assessment with available empirical evidence from publicly disclosed vulnerabilities, CWE is developing a stronger evidence base for hardware weakness prioritization while retaining the specialized judgment needed to interpret a comparatively limited and uneven body of public data.

Hardware work has also influenced CWE's broader architecture. It exposed limitations in lifecycle terminology and highlighted the need to distinguish formal weakness relationships from groupings created for a particular community. Those lessons are informing CWE 5.0 planning.

From expansion to a richer model

2020	→	Hardware enters CWE through HW SIG expanding content
2021 +	→	MIHW prioritizes hardware weaknesses
CWE 5.0	→	Lifecycle and structure lessons inform the next architecture

“

The Hardware CWE revolutionized semiconductor security assurance by codifying tribal knowledge into structured, actionable guidance for design, verification, and PSIRT communities. It inspires academic research and EDA innovations to address the biggest challenges impacting our industry. Today, this standardized taxonomy serves as the foundational ground truth for training next-generation, security-aware LLMs in the era of AI.

Jason Fung

Senior Director, Offensive Security
Research & Academic Research
Engagement, Intel Corporation
CWE Board member



Twenty Years of Evolution

A parallel view of how CWE evolved as a knowledge base and how its use spread across tools, standards, vulnerability programs, and security practice.

CWE EVOLUTION

How the knowledge base changed

2006 CWE becomes public

First public drafts and a dedicated website establish a common language for software weaknesses.

2008 Structure matures; CWE 1.0

Named chains, composites, views, and richer relationships take shape; CWE 1.0 establishes a more stable foundation later that year.

2009 - 2011 Prioritization becomes systematic

The first CWE Top 25 is followed by CWE 2.0, CWSS, and CWRAP, extending CWE from broad prioritization toward scoring and context-specific weakness risk analysis.

2014 External knowledge enriches CWE

CWE 2.8 incorporates Software Fault Patterns and SOAR research, expanding mappings, detection methods, and relationships.

2017 CWE 3.0 modernizes the model

Schema 6.0 and the Architectural Concepts view refine how weakness concepts and relationships are represented.

2019 New knowledge + data-driven analysis

CWE 3.2 incorporates CQE-derived quality content; the Top 25 returns with a data-driven methodology.

2020 - 2021 Hardware enters CWE

CWE 4.0 formally adds hardware design weaknesses; the first Most Important Hardware Weaknesses list follows in 2021.

2023 - 2026 Building for the next era

CDR-based content development, AI/ML coverage, RTL work, improved mapping guidance, more clear and consumable website pages, and CWE 5.0 planning broaden how CWE is developed and used.

ECOSYSTEM ADOPTION

How the community put CWE to work

2006 - 2024 Compatibility Program scales adoption

Created to promote consistent use of CWE in security tools, services, and development practices.

2007 7 organizations • 15 product/service declarations

2011 First 13 products formally certified CWE-Compatible – 5 organizations across 3 countries

2024 87 organizations • 148 products & services • 20 countries • 104 officially CWE-Compatible

Program listings archived in April 2024; current compatibility requirements remain available for self-attestation.

SUMMER 2007 CWE enters NVD operations

NIST begins using CWE to classify vulnerabilities in the National Vulnerability Database, making weakness information part of a major public vulnerability-data service.

2010 - 2021 CWE spreads to standards and guidance

2010 OWASP Top 10 – maps major web-application risks to CWE

2012 ITU-T X.1524 – establishes an international recommendation specifically for CWE

2012/15 ISO/IEC TR 20004 – uses CWE in Common Criteria vulnerability analysis

2013 ISO/IEC TR 24772 – uses CWE identifiers in programming-language vulnerability guidance

2014 ITU-T X.1208 + ISO/IEC 29147 – apply CWE in risk and vulnerability-disclosure guidance

2021 ISO/IEC 5055 – uses selected CWE weaknesses in automated source-code quality measures

2020s Hardware verification adopts CWE

Hardware-security and EDA/pre-silicon verification tools begin using CWE to map automated checks, define security requirements, and report coverage against hardware weakness classes.

MAY 2024 CNA Rules strengthen CWE linkage

CVE CNA Operational Rules 4.0 require CVE Records to identify vulnerability type and recommend using CWE to classify the type or cause.

2024 Machine access expands

The public REST API makes CWE easier to consume programmatically in downstream tools, services, and automated workflows.

CWE at 20

944

CURRENT WEAKNESSES
CWE 4.20

1,450

TOTAL CWE ENTRIES
Weaknesses + Categories + Views +
Preserved Deprecated Entries

39,080

CVE RECORDS
Included in the
2025 Top 25 analysis

555

DISTINCT CWE IDs REPRESENTED
IN THE 2025 TOP 25 ANALYSIS
across 50,681 CWE mappings

Corpus growth

609	706	839	934	944
weaknesses	weaknesses	weaknesses	weaknesses	weaknesses
2008	2014	2020	2023	2026

GROWTH ALSO HAPPENED *INSIDE* THE ENTRIES

Entry counts do not capture the additional demonstrative and observed examples, mapping guidance, detection methods, mitigations, relationships, and domain-specific content now carried in the corpus. Recent modernization work also measures completeness so improvement can be directed toward the information users need.

CWE'S REACH GREW WITH THE KNOWLEDGE BASE

76x

GROWTH IN DIGITAL REACH
7.7K → 582K

August 2008 → August 2026
unique IP addresses observed on cwe.mitre.org
2008 7.7K → 2011 31.5K → 2017 70.5K
2020 146.6K → 2026 582K

86.4M

REST API REQUESTS
September 2025 – August 2026
18.4K unique IP addresses

accessed the API in August 2026
4.7x the August 2025 level

Server-log measures. Unique IP addresses are a proxy for reach, not unique individual users; request totals may include automated traffic.

From Knowledge to Practice

A shared weakness language becomes most useful when it enters practice. Over two decades, CWE has been used in secure-coding and software-assurance education, development guidance, and standards that help organizations describe, assess, and address recurring weaknesses consistently.

These uses carry weakness knowledge from the corpus into the decisions people make while learning, building, and evaluating technology.

HOW WEAKNESS KNOWLEDGE MOVES INTO PRACTICE

LEARN

Education & training use CWE to teach recurring weakness patterns, connect concepts to real examples, and show where weaknesses can enter the development lifecycle.

BUILD

Development guidance can map coding rules and recommendations to CWE, linking engineering practices to the weaknesses they are intended to prevent.

ASSURE

Standards & assurance frameworks use CWE as a common reference for vulnerability analysis, software measurement, and interoperability across tools and assessments.

SELECTED EXAMPLES FROM ACROSS THE ECOSYSTEM

EARLY EDUCATION USE

Purdue University / CERIAS — 2007 Work on *Integrating the Common Weakness Enumeration into a Secure Programming Course* demonstrated how CWE could be organized and applied directly in secure-programming education.

Secure coding • Software assurance • Vulnerability analysis

CWE IN THE CLASSROOM TODAY

University courses across the globe use CWE in secure coding, software security, vulnerability analysis, and related cybersecurity education. A selection includes:

Carnegie Mellon Univ. • Stanford • Cornell • Univ. of Washington • Harvard • UC Berkley • UTEP • Univ. of Nebraska at Omaha • Northeastern • RIT • Univ. of Maryland • Univ. of Edinburgh • Univ. of Tasmania • Amrita Vishwa Vidyapeetham

SECURE DEVELOPMENT GUIDANCE

CWE mappings connect secure coding rules and guidance to the CWEs that those practices are intended to address.

SEI CERT Secure Coding Standards for C, C++, Java, Perl, and Android™ • OpenSSF Secure Coding Guide for Python • SAFEcode • the OWASP Application Security Verification Standard • BSA Framework for Secure Software

INTERNATIONAL STANDARDS & ASSURANCE

CWE is used in international standards addressing weakness interoperability, Common Criteria vulnerability analysis, and automated source-code quality measurement. A selection includes:

ITU-T X.1524 • ISO/IEC TR 20004 • ISO/IEC 5055 • ETSI GS ISI 002 V1.2.1. • CISQ quality/data-protection measures

A COMMON LANGUAGE CAN CHANGE PRACTICE

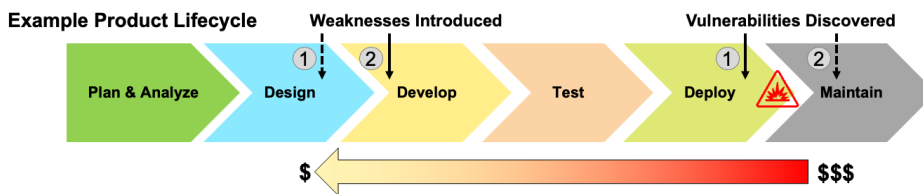
Education helps people recognize recurring weakness patterns. Guidance connects those patterns to development decisions. Standards make the vocabulary portable across organizations, tools, and assessments. Together, these move weakness knowledge closer to where recurring problems can be prevented.

From Root Cause to Prevention

A fixed vulnerability can still inform prevention.

When vulnerabilities are mapped to the weaknesses that caused them, recurring CWE patterns can show where similar mistakes continue to enter products. Those patterns may point to architecture changes, implementation practices, verification gaps, technology choices, or organizational processes that deserve attention. Tooling may help detect and identify weaknesses at different times as well.

The lesson becomes useful when it returns to product development, as avoiding weaknesses earlier in the lifecycle is easier and less costly.



CISA's Secure by Design work encourages manufacturers to take greater responsibility for security outcomes and demonstrate measurable progress in reducing vulnerability classes. Root-cause information can help organizations understand which weakness classes continue to appear and whether changes in development are altering the pattern.

CWE provides a shared vocabulary for that analysis. The objective is not to replace vulnerability management; it is to learn from vulnerability information in a way that can influence future engineering.

VULNERABILITY MAPPING GUIDANCE, BUILT INTO CWE

Since CWE v4.14, every CWE entry has included a vulnerability mapping label under its name, linking directly to detailed guidance.

1 VULNERABILITY DISCOVERED

Capture the product-level evidence.

2 IDENTIFY ROOT CAUSE

Map the available evidence to an appropriate CWE.

3 LOOK FOR RECURRENCE

Examine patterns across products and time.

4 CHANGE DEVELOPMENT

Feed into architecture, implementation, verification, technology, or process.

5 MEASURE WHAT HAPPENS NEXT

Does this weakness keep appearing? Where does it enter the lifecycle? Can we detect it earlier? What intervention could reduce it? Are our products showing improvement?

“

CWE helps transform vulnerability information into knowledge that can drive prevention. CISA is proud to sponsor a resource that gives the community a common language for describing the weaknesses behind vulnerabilities. Taking that knowledge upstream helps product developers recognize the weaknesses that recur in their products and apply Secure by Design practices to reduce those weakness classes over time.

Lindsey Cerkovnik

CISA Vulnerability Response and Coordination, Chief

Current CISA Sponsor Representative for CWE

Making CWE Easier to Apply

A common language works best when users can find the right concept, interpret it consistently, and incorporate the information into their own processes and tools.

Recent CWE modernization has focused on those practical needs. Expanded mapping guidance helps vulnerability analysts avoid concepts that are too abstract or otherwise inappropriate for root-cause mapping. Content improvements have added examples, detection methods, mitigation information, weakness visualizations, content filtering capabilities, and clearer descriptions.

Supplier CVE Numbering Authorities (CNAs) , being closest to the affected products and underlying vulnerability information, are best positioned to identify the appropriate CWE(s) and support more precise root-cause mapping.	From 2021 to 2022 , 39% of CVE Records included a CNA-published CWE Mapping	From 2025 to 2026 , 77% of CVE Records included a CNA-published CWE mapping
---	---	---

The increase coincided with greater emphasis on first-party mapping across vulnerability management. CVE’s CNA Operational Rules now require Records to specify a vulnerability type or cause and recommend CWE for doing so, while CWE mapping guidance and direct engagement with CNAs have placed greater attention on mapping accuracy and specificity.

Machine access has grown as well. The public CWE REST API, launched in 2024, supports programmatic use of current CWE information. As more tools and analytical workflows consume CWE directly, testing and verification of API-served data have become increasingly important.

The Content Development Repository (CDR) gives contributors a structured way to propose and follow changes while preserving expert technical review. The Program has also begun evaluating content completeness more systematically to direct effort toward information users need.

Recent work distinguishes breadth – whether expected content elements are present – from depth, which asks whether those elements contain enough useful technical detail.





Built with the Community

Working groups, Special Interest Groups, and individual contributors bring specialized expertise into CWE as technologies and community priorities change.

Community collaboration has shaped CWE from the beginning. The work that led to CWE grew through efforts such as NIST's SAMATE project, bringing together PLOVER and knowledge from industry, government, academia, and other software-security sources to build a common foundation for describing weaknesses.

That pattern continued as CWE matured. Contributions and external bodies of knowledge – including Software Fault Patterns – expanded and refined the corpus. Today, the CWE Board, working groups, special interest groups, and the Content Development Repository bring new expertise and public contributions into the Program as technologies and security needs change.

CWE COMMUNITIES FORM AROUND THE WORK

● **CURRENT / ACTIVE IN 2026** — **RETIRED/EMAIL-ONLY** ○ **TRANSITIONING** ◆ **PLANNED**

2020–2021. A broader community model takes shape

- **CWE Board** — cross-cutting technical and strategic guidance
- **Hardware CWE SIG** — hardware design, verification, and weakness expertise
- **User Experience WG** — user needs, presentation, guidance, and usability work; est. 2021

2022–2024. Groups form around specific needs

- **ICS/OT SIG** — Domain coverage and mappings to ISA/IEC 62443 in partnership with the Securing Energy Infrastructure Executive Task Force (SEI ETF)
- **REST API WG** — Requirements for programmatic access to CWE data
- **Root Cause Mapping WG** — More accurate, scalable CVE-to-CWE mapping and guidance

2025–2026. Specialized expertise deepens

- **AI Working Group** — Systematic assessment of AI/ML weakness coverage and gaps
- **RTL Ad Hoc Working Group** — Specialized content development at register-transfer level

LOOKING AHEAD - Community mechanisms continue to evolve

- ◆ **Cross-Domain SIG** — **PLANNED** — Intended to mix usability, root cause mapping, SDLC integration, and other cross-cutting applications into a regular forum

COMMUNITY INPUT BECOMES CWE

- DOMAIN EXPERTISE** → New and improved weakness content
- USER + TOOL NEEDS** → Guidance, navigation, presentation, and API capabilities
- MAPPING EXPERTISE** → More precise root cause mapping use and weakness analysis
- PUBLIC CONTRIBUTIONS** → Content Development Repository (CDR), traceability, and corpus changes

"CWE's strength comes from community collaboration, where experts across various domains bring their experience and perspectives to strengthen and expand the knowledge base. We saw the value of that model as CWE expanded into hardware, and we are seeing it again as the community enumerates weaknesses unique to AI/ML technologies. Bringing together expertise from across disciplines helps identify gaps, refine concepts, and ensure CWE continues to reflect technologies and security challenges as they evolve."

Jason Oberg

Fellow, Security Solutions
Arteris, Inc.
CWE Board member





AI Changes the Landscape Twice

AI AS A DOMAIN

CWE 4.20 introduced a dedicated AI/ML view that organizes both weaknesses specific to AI/ML technology and established software weaknesses that remain relevant in AI-enabled products. Many reported AI-related vulnerabilities still trace to familiar software weakness classes.

The challenge is therefore to identify genuinely new weakness concepts without recreating existing ones simply because a familiar security problem appears in an AI context. The CWE AI Working Group is increasingly examining coverage systematically to determine where existing CWE knowledge applies, where entries need additional AI-specific context, and where genuine gaps remain.

ESTABLISHED WEAKNESSES

Existing CWE concepts applied in AI-enabled products

AI/ML-SPECIFIC WEAKNESSES

New concepts where AI/ML introduces genuinely distinct weakness patterns



AI AS AN ENABLER

CWE is also an input to AI-enabled security analysis. Its curated definitions, relationships, examples, and mapping guidance provide structured reference knowledge that can help ground AI systems used for retrieval, explanation, and vulnerability-to-CWE mapping.

Recent CWE experiments have explored the other side of that relationship: using large language models to help analysts find relevant CWE information and identify candidate mappings. Early results showed useful capabilities, but also overly confident suggestions, misuse of evidence, and mappings where the available information was insufficient.

The emerging model is complementary: CWE provides structured weakness knowledge; AI can make that knowledge easier to retrieve and analyze; experts remain responsible for validating conclusions.

CURATED CWE KNOWLEDGE

Definitions + relationships + examples + guidance



AI-Assisted Analysis

Definitions + relationships + examples + guidance



Expert Judgment

Evidence + validation + decision

CURRENT WORKING PRINCIPLE

AI can extend access to CWE knowledge, while CWE can help ground AI-assisted security analysis. Reliable root-cause mapping still depends on sufficient evidence, current CWE information, appropriate confidence, and expert judgment.



CWE 5.0: Building for the Next Era

CWE's structure has evolved throughout its history as the knowledge base has grown and new uses have emerged. CWE 5.0 continues that progression, revisiting parts of the underlying model to support a broader range of technologies, development practices, and users.

The need is broader than adding new weakness content. Software development and vulnerability management are changing quickly: AI is influencing how software is created and analyzed, modern products depend on increasingly complex components and supply chains, and vulnerability information is being produced and consumed at greater scale. These shifts increase the need for precise root-cause information and weakness knowledge that developers, analysts, tools, and automated systems can interpret and apply.

CWE 5.0 is designed around two practical priorities:

More precise root-cause mapping | Better support for developers learning to prevent weaknesses.

CLEARER CONCEPTS

Clearer terminology helps reduce ambiguity for people mapping, teaching, and applying CWE information for different use cases.

CLEARER RELATIONSHIPS

Explicit relationship dimensions can make the structure easier to interpret, maintain, and analyze for gaps or overlap.

BROADER LIFECYCLES

More flexible lifecycle models can represent where weaknesses arise and are mitigated across software, hardware, and AI/ML development.

BETTER STRUCTURED CONTENT

Purpose-specific technical elements can make CWE easier to understand, process automatically, and incorporate into emerging analytical workflows.

LANGUAGE CHANGES THE MENTAL MODEL.

Even the Names Matter.

Much of cybersecurity practice has traditionally been organized around attacks and vulnerabilities, which can make CWE's weakness-oriented language less intuitive to many users. But terminology influences how a problem is understood – and where people look for solutions.

CWE 5.0 is revisiting attack-oriented names where appropriate, preserving familiar terminology as Alternate Terms while making the underlying weakness clearer. A clearer weakness vocabulary can help users move from describing what happened toward understanding why it was possible and how recurrence might be prevented.



The Next 20 Years

CWE enters its third decade as the way software is produced is changing rapidly. AI-assisted development is already generating code at increasing scale, while software continues to incorporate hardware, cloud services, open source components, and AI-enabled capabilities. Developers may increasingly describe intent while automated systems generate, test, and modify implementations.

That shift creates a new challenge for security. The amount of code that can be produced may exceed what people can review directly, increasing the importance of automated analysis backed by well-structured security knowledge. AI agents may also help analyze code, identify weakness patterns, and communicate threat models – but those capabilities depend on consistent concepts and terminology that people and machines can interpret.

CWE should be part of that foundation. A shared weakness language can connect vulnerabilities, code analysis, threat modeling, testing, and development decisions around the recurring conditions that make systems vulnerable. As development becomes more automated, names and definitions for those conditions become more important.

The opportunity for CWE's next era is to help make prevention more systematic: learning from weaknesses that appear in deployed products, carrying those lessons into increasingly automated development environments, and measuring whether recurring weakness classes are actually declining.

Three Actions for the Next Era

1. FEED THE LESSONS BACK

Carry weakness information into human-/AI-assisted product development.

2. MEASURE RECURRENCE

Know which weakness types persist and whether interventions are changing the trend.

3. OWN THE OUTCOME

Make reduction of preventable weakness classes a product-security objective.

CWE was built by a community – and continues to evolve because of it. Thank you to all the sponsors, researchers, developers, practitioners, educators, tool providers, leaders, champions, and contributors who have shaped CWE over the last twenty years.

“

We are entering the fifth industrial revolution, the intelligence age... As CVE disclosures increase and time-to-exploit decreases, CWE must continue to evolve to capture new weakness types and contexts, reason about weakness chains, and be accurately cited and consumed by both vulnerability disclosures and AI. Without precise terminology and reasoning, AI-driven automation risks amplifying mistakes and wasted effort... CWE is an enabler of automation.

ALEXANDER HOOLE

Senior Manager,
Software Security Research,
OpenText Fortify,
and CWE Board Member



About CWE

The Common Weakness Enumeration (CWE™) is a community-developed knowledge base of software and hardware weakness types. CWE provides a common language for describing conditions that can contribute to vulnerabilities and supports cybersecurity analysis, development, research, education, and related uses.

cwe.mitre.org

Terms of Use and Copyright

CWE is free to use for research, development, and commercial purposes under a non-exclusive, royalty-free license from The MITRE Corporation. Copies are authorized subject to reproduction of MITRE's copyright designation and applicable license terms. Full terms and disclaimers: cwe.mitre.org/about/termsofuse.html

Copyright © 2006–2026, The MITRE Corporation.

CWE, CWSS, CWRAF, and the CWE logo are trademarks of The MITRE Corporation.

The CWE Program is sponsored by CISA and managed through the Homeland Security Systems Engineering and Development Institute (HSSEDI), operated by The MITRE Corporation.