

CWE Top 25 Macro Samples

61CWE Top 25 Issues

13CWE Top 25 Hotspots

1d 6h

CWE Top 25 Remediation Effort

E

CWE Top 25 Rating

≡

CWE/SANS Top 25 Most Dangerous Software Weaknesses 2022

RANK	ID	NAME	RATING	ISSUES	HOTSPOTS (3)
> [1]	CWE-787	Out-of-bounds Write	5	0	0
> [2]	CWE-79	Improper Neutralization of Input During Web Page Generation (Cross-site Scripting)	5	0	0
> [3]	CWE-48	Improper Neutralization of Special Elements used in an SQL Command (SQL Injection)	5	0	0
> [4]	CWE-20	Improper Input Validation	5	14	0
> [5]	CWE-128	Out-of-bounds Read	5	0	0
> [6]	CWE-78	Improper Neutralization of Special Elements used in an OS Command (OS Command Injection)	5	0	0
> [7]	CWE-419	Use After Free	5	0	0
> [8]	CWE-52	Improper Limitation of a Pathname to a Restricted Directory (Path Traversal)	5	0	0
> [9]	CWE-352	Cross-Site Request Forgery (CSRF)	5	0	0
> [10]	CWE-434	Unrestricted Upload of File with Dangerous Type	5	0	0
> [11]	CWE-475	NULL Pointer Dereference	5	48	0
> [12]	CWE-502	Deserialization of Untrusted Data	5	0	0
> [13]	CWE-190	Integer Overflow or Wraparound	5	0	0
> [14]	CWE-287	Improper Authentication	5	0	0
> [15]	CWE-788	Use of Hard-coded Credentials	5	0	0
> [16]	CWE-882	Missing Authorization	5	0	0
> [17]	CWE-77	Improper Neutralization of Special Elements used in a Command (Command Injection)	5	0	0
> [18]	CWE-358	Missing Authentication for Critical Function	5	0	0
> [19]	CWE-119	Improper Restriction of Operations within the Bounds of a Memory Buffer	5	0	0
> [20]	CWE-276	Incorrect Default Permissions	5	0	0
> [21]	CWE-353	Server-Side Request Forgery (SSRF)	5	0	0
> [22]	CWE-362	Concurrent Execution using Shared Resource with Improper Synchronization (Race Condition)	5	0	0
> [23]	CWE-400	Uncontrolled Resource Consumption	5	0	13
> [24]	CWE-811	Improper Restriction of XML External Entity Reference	4	1	0
> [25]	CWE-94	Improper Control of Generation of Code (Code Injection)	5	0	0

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

61

CWE Top 25 Macro Samples

CWE Top 25 Macro Multi project

2 projects

74 CWE Top 25 Issues

18 CWE Top 25 Hotspots

2d 1h
CWE Top 25 Remediation Effort

E
CWE Top 25 Rating

CWE/SANS Top 25 Most Dangerous Software Weaknesses 2022

RANK	ID	NAME	RATING	ISSUES	HOTSPOTS (3)
> [1]	CWE-767	Out-of-bounds Write	✓	0	0
> [2]	CWE-79	Improper Neutralization of Input During Web Page Generation (Cross-site Scripting)	✓	0	0
> [3]	CWE-88	Improper Neutralization of Special Elements used in an SQL Command (SQL Injection)	✓	0	0
▼ [4]	CWE-20	Improper Input Validation	✓	14	0

CWE-20 Improper Input Validation

The product receives input or data, but it does not validate or incorrectly validates that the input has the properties that are required to process the data safely and correctly.

Input validation is a frequently-used technique for checking potentially dangerous inputs in order to ensure that the inputs are safe for processing within the code, or when communicating with other components. When software does not validate input properly, an attacker is able to craft the input in a form that is not expected by the rest of the application. This will lead to parts of the system receiving unintended input, which may result in altered control flow, arbitrary control of a resource, or arbitrary code execution. Input validation is not the only technique for processing input, however. Other techniques attempt to transform potentially dangerous input into something safe, such as filtering (CWE-767), which attempts to remove dangerous input - or encoding/escaping (CWE-778), which attempts to ensure that the input is not misinterpreted when it is included in output to another component. Other techniques exist as well (see CWE-538 for more examples).

Severity	Rule name	Type	CWE_ID	Total
✓	Logging should not be vulnerable to injection attacks	✓	CWE-20	14

> [5]	CWE-128	Out-of-bounds Read	✓	0	0
> [6]	CWE-78	Improper Neutralization of Special Elements used in an OS Command (OS Command Injection)	✓	0	0
> [7]	CWE-416	Use After Free	✓	0	0
> [8]	CWE-122	Improper Limitation of a Pathname to a Restricted Directory (Path Traversal)	✓	0	0
> [9]	CWE-352	Cross-Site Request Forgery (CSRF)	✓	0	0
> [10]	CWE-434	Unrestricted Upload of File with Dangerous Type	✓	0	0
▼ [11]	CWE-476	NULL Pointer Dereference	✓	31	0

CWE-476 NULL Pointer Dereference

A NULL pointer dereference occurs when the application dereferences a pointer that it expects to be valid, but is NULL, typically causing a crash or exit.

NULL pointer dereference issues can occur through a number of ways, including race conditions, and simple programming mistakes.

Severity	Rule name	Type	CWE_ID	Total
✓	Null should not be returned from a "boolean" method	✓	CWE-476	1
✓	Null pointers should not be dereferenced	✓	CWE-476	30

> [12]	CWE-352	Deserialization of Untrusted Data	✓	0	0
> [13]	CWE-190	Integer Overflow or Wraparound	✓	0	0
> [14]	CWE-287	Improper Authentication	✓	0	0
> [15]	CWE-768	Use of Hard-coded Credentials	✓	0	0

