MSC24-C. Do not use deprecated or obsolescent functions

Do not use deprecated or obsolescent functions when more secure equivalent functions are available. Deprecated functions are defined by the C Standard. Obsolescent functions are defined by this recommendation.

Deprecated Functions

The `gets()` function was deprecated by Technical Corrigendum 3 to C99 and eliminated from C11.

Obsolescent Functions

Functions in the first column of the following table are hereby defined to be *obsolescent functions*. To remediate invocations of obsolescent functions, an application might use inline coding that, in all respects, conforms to this guideline, or an alternative library that, in all respects, conforms to this guideline, or alternative *non-obsolescent functions*.

Obsolescent Function	Recommended Alternative	Rationale
`asctime()`	`asctime_s()`	Non-reentrant
`atof()`	`strtod()`	No error detection
`atoi()`	`strtol()`	No error detection
`atol()`	`strtol()`	No error detection
`atoll()`	`strtoll()`	No error detection
`ctime()`	`ctime_s()`	Non-reentrant
`fopen()`	`fopen_s()`	No exclusive access to file
`freopen()`	`freopen_s()`	No exclusive access to file
`rewind()`	`fseek()`	No error detection
`setbuf()`	`setvbuf()`	No error detection

The `atof()`, `atoi()`, `atol()`, and `atoll()` functions are obsolescent because the `strtod()`, `strtof()`, `strtol()`, `strtold()`, `strtoll()`, `strtoul()`, and `strtoull()` functions can emulate their usage and have more robust error handling capabilities. See INT05-C. Do not use input functions to convert character data if they cannot handle all possible inputs.

The `fopen()` and `freopen()` functions are obsolescent because the `fopen_s()` and `freopen_s()` functions can emulate their usage and improve security by protecting the file from unauthorized access by setting its file protection and opening the file with exclusive access [ISO/IEC WG14 N1173].

The `setbuf()` function is obsolescent because `setbuf()` does not return a value and can be emulated using `setvbuf()`. See ERR07-C. Prefer functions that support error checking over equivalent functions that don't.

The `rewind()` function is obsolescent because `rewind()` does not return a value and can be emulated using `fseek()`. See ERR07-C. Prefer functions that support error checking over equivalent functions that don't.

The `asctime()` and `ctime()` functions are obsolescent because they use non-reentrant static buffers and can be emulated using `asctime_s()` and `ctime_s()`.

Unchecked Obsolescent Functions

The following are hereby defined to be *unchecked obsolescent functions*:

	`bsearch`		`fprintf`	`fscanf`	`fwprintf`	`fwscanf`
`getenv`	`gmtime`	`localtime`	`mbsrtowcs`	`mbstowcs`	`memcpy`	`memmove`
`printf`	`qsort`	`setbuf`	`snprintf`	`sprintf`	`sscanf`	`strcat`
`strcpy`	`strerror`	`strncat`	`strncpy`	`strtok`	`swprintf`	`swscanf`
`vfprintf`	`vfscanf`	`vfwprintf`	`vfwscanf`	`vprintf`	`vscanf`	`vsnprintf`
`vsprintf`	`vsscanf`	`vswprintf`	`vswscanf`	`vwprintf`	`vwscanf`	`wcrtomb`
`wcscat`	`wcscpy`	`wcsncat`	`wcsncpy`	`wcsrtombs`	`wcstok`	`wcstombs`
`wctomb`	`wmemcpy`	`wmemmove`	`wprintf`	`wscanf`		

To remediate invocations of unchecked obsolescent functions, an application might use inline coding that, in all respects, conforms to this guideline, or an alternative library that, in all respects, conforms to this guideline, or alternative *nonobsolescent functions* from C11, Annex K:

abort_handler_s		bsearch_s		fprintf_s	freopen_s	fscanf_s
fwprintf_s	fwscanf_s	getenv_s	gets_s	gmtime_s	ignore_handler_s	localtime_s
mbsrtowcs_s	mbstowcs_s	memcpy_s	memmove_s	printf_s	qsort_s	scanf_s
set_constraint_handler_s	snprintf_s	snwprintf_s	sprintf_s	sscanf_s	strcat_s	strcpy_s
strerror_s	strerrorlen_s	strncat_s	strncpy_s	strnlen_s	strtok_s	swprintf_s
swscanf_s	vfprintf_s	vfscanf_s	vfwprintf_s	vfwscanf_s	vprintf_s	vscanf_s
vsnprintf_s	vsnwprintf_s	vsprintf_s	vsscanf_s	vswprintf_s	vswscanf_s	vwprintf_s
vwscanf_s	wcrtomb_s	wcrtoms_s	wcscat_s	wcscpy_s	wcsncat_s	wcsncpy_s
wcsnlen_s	wcsrtombs_s	wcstok_s	wcstombs_s	wctomb_s	wmemcpy_s	wmemmove_s
wprintf_s	wscanf_s					

or alternative *nonobsolescent functions* from ISO/IEC TR 24731-2, *Extensions to the C Library—Part II: Dynamic Allocation Functions* [ISO/IEC TR 24731-2]:

asprintf	aswprintf	fmemopen	fscanf	fwscanf	getdelim	getline
getwdelim	getwline	open_memstream	open_wmemstream	strdup	strndup	

Noncompliant Code Example

In this noncompliant code example, the obsolescent functions `strcat()` and `strcpy()` are used:

```
#include <string.h>
#include <stdio.h>

enum { BUFSIZE = 32 };
void complain(const char *msg) {

  static const char prefix[] = "Error: ";
  static const char suffix[] = "\n";
  char buf[BUFSIZE];

  strcpy(buf, prefix);
  strcat(buf, msg);
  strcat(buf, suffix);
  fputs(buf, stderr);
}
```

Compliant Solution

In this compliant solution, `strcat()` and `strcpy()` are replaced by `strcat_s()` and `strcpy_s()`:

```
#define __STDC_WANT_LIB_EXT1__
#include <string.h>
#include <stdio.h>

enum { BUFFERSIZE = 256 };

void complain(const char *msg) {
  static const char prefix[] = "Error: ";
  static const char suffix[] = "\n";
  char buf[BUFFERSIZE];

  strcpy_s(buf, BUFFERSIZE, prefix);
  strcat_s(buf, BUFFERSIZE, msg);
  strcat_s(buf, BUFFERSIZE, suffix);
  fputs(buf, stderr);
}
```

Risk Assessment

The deprecated and obsolescent functions enumerated in this guideline are commonly associated with software vulnerabilities.

Rule	Severity	Likelihood	Remediation Cost	Priority	Level
MSC24-C	High	Probable	Medium	P12	L1

Automated Detection

Tool	Version	Checker	Description
Astrée	19.04	**stdlib-use-ato** **stdlib-macro-ato** **stdlib-use-atoll** **stdlib-macro-atoll**	Partially checked
Axivion Bauhaus Suite	6.9.0	**CertC-MSC24**	Fully implemented
CodeSonar	5.2p0	**BADFUNC.*** **(customization)**	A number of CodeSonar's "Use of *" checks are for deprecated/obsolescent functions CodeSonar also provides a mechanism for users to create custom checks for uses of specified functions
ECLAIR	1.2	**CC2.MSC34**	Fully implemented
LDRA tool suite	9.7.1	**44 S**	Fully implemented
Parasoft C/C++test	10.4.2	**CERT_C-MSC24-a** **CERT_C-MSC24-b** **CERT_C-MSC24-c** **CERT_C-MSC24-d**	The library functions atof, atoi and atol from library stdlib.h shall not be used The library functions abort, exit, getenv and system from library stdlib.h shall not be used Avoid using unsafe string functions which may cause buffer overflows Don't use unsafe C functions that do write to range-unchecked buffers
Polyspace Bug Finder	R2019b	CERT C: Rec. MSC24-C	Checks for use of obsolete standard function (rec. fully covered)
PVS-Studio	6.23	**V513**, **V2001**, **V2002**	
RuleChecker	19.04	**stdlib-use-ato** **stdlib-macro-ato** **stdlib-use-atoll** **stdlib-macro-atoll**	Partially checked

Related Vulnerabilities

Search for vulnerabilities resulting from the violation of this rule on the CERT website.

Related Guidelines

CERT C Secure Coding Standard	ERR07-C. Prefer functions that support error checking over equivalent functions that don't INT05-C. Do not use input functions to convert character data if they cannot handle all possible inputs ERR34-C. Detect errors when converting a string to a number STR06-C. Do not assume that strtok() leaves the parse string unchanged STR07-C. Use the bounds-checking interfaces for string manipulation
ISO/IEC TR 24772	Use of Libraries [TRJ]
MISRA C:2012	Rule 21.3 (required)
MITRE CWE	CWE-20, Insufficient input validation CWE-73, External control of file name or path CWE-79, Improper Neutralization of Input During Web Page Generation ('Cross-site Scripting') CWE-89, Improper Neutralization of Special Elements used in an SQL Command ('SQL Injection') CWE-91, XML Injection (aka Blind XPath Injection) CWE-94, Improper Control of Generation of Code ('Code Injection') CWE-114, Process Control CWE-120, Buffer Copy without Checking Size of Input ('Classic Buffer Overflow') CWE-192, Integer coercion error CWE-197, Numeric truncation error CWE-367, Time-of-check, time-of-use race condition CWE-464, Addition of data structure sentinel CWE-601, URL Redirection to Untrusted Site ('Open Redirect') CWE-676, Use of potentially dangerous function

Bibliography

[Apple 2006]	Apple Secure Coding Guide, "Avoiding Race Conditions and Insecure File Operations"
[Burch 2006]	Specifications for Managed Strings, Second Edition
[Drepper 2006]	Section 2.2.1 "Identification When Opening"
[IEEE Std 1003.1:2013]	XSH, System Interfaces, `open`
ISO/IEC 23360-1:2006	
[ISO/IEC WG14 N1173]	Rationale for TR 24731 Extensions to the C Library Part I: Bounds-checking interfaces
[Klein 2002]	"Bullet Proof Integer Input Using `strtol()`"
[Linux 2008]	`strtok(3)`
[Seacord 2013]	Chapter 2, "Strings" Chapter 8, "File I/O"
[Seacord 2005b]	"Managed String Library for C, C/C++"