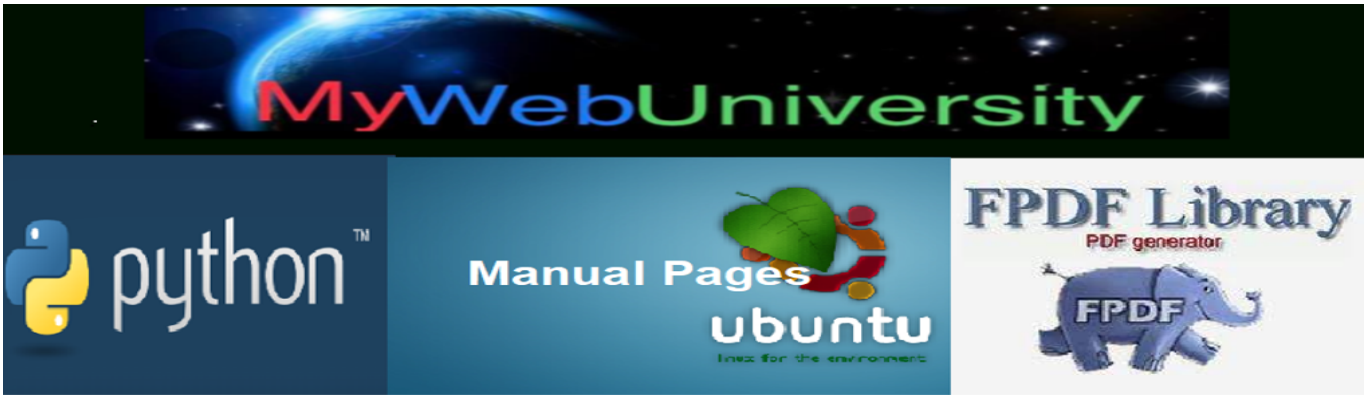

Rocky Enterprise Linux 9.2 Manual Pages on command 'signal-safety.7'

$ man signal-safety.7

SIGNAL-SAFETY(7) Linux Programmer's Manual SIGNAL-SAFETY(7)

NAME

signal-safety - async-signal-safe functions

DESCRIPTION

An async-signal-safe function is one that can be safely called from within a signal han? dler. Many functions are not async-signal-safe. In particular, nonreentrant functions are generally unsafe to call from a signal handler.

The kinds of issues that render a function unsafe can be quickly understood when one con? siders the implementation of the stdio library, all of whose functions are not async-sig? nal-safe.

When performing buffered I/O on a file, the stdio functions must maintain a statically al? located data buffer along with associated counters and indexes (or pointers) that record the amount of data and the current position in the buffer. Suppose that the main program is in the middle of a call to a stdio function such as printf(3) where the buffer and as? sociated variables have been partially updated. If, at that moment, the program is inter? rupted by a signal handler that also calls printf(3), then the second call to printf(3) will operate on inconsistent data, with unpredictable results.

To avoid problems with unsafe functions, there are two possible choices:

1. Ensure that (a) the signal handler calls only async-signal-safe functions, and (b) the signal handler itself is reentrant with respect to global variables in the main pro? gram.
2. Block signal delivery in the main program when calling functions that are unsafe or op? erating on global data that is also accessed by the signal handler.

Generally, the second choice is difficult in programs of any complexity, so the first choice is taken.

POSIX.1 specifies a set of functions that an implementation must make async-signal-safe. (An implementation may provide safe implementations of additional functions, but this is not required by the standard and other implementations may not provide the same guaran? tees.)

In general, a function is async-signal-safe either because it is reentrant or because it is atomic with respect to signals (i.e., its execution can't be interrupted by a signal handler).

The set of functions required to be async-signal-safe by POSIX.1 is shown in the following table. The functions not otherwise noted were required to be async-signal-safe in POSIX.1-2001; the table details changes in the subsequent standards.

Function	Notes
abort(3)	Added in POSIX.1-2001 TC1
accept(2)	
access(2)	
aio_error(3)	
aio_return(3)	
aio_suspend(3)	See notes below
alarm(2)	
bind(2)	
cfgetispeed(3)	
cfgetospeed(3)	
cfsetispeed(3)	
cfsetospeed(3)	
chdir(2)	
chmod(2)	
chown(2)	
clock_gettime(2)	
close(2)	
connect(2)	
creat(2)	
dup(2)	

dup2(2)

execl(3) Added in POSIX.1-2008; see notes below

execle(3) See notes below

execv(3) Added in POSIX.1-2008

execve(2)

_exit(2)

_Exit(2)

faccessat(2) Added in POSIX.1-2008

fchdir(2) Added in POSIX.1-2008 TC1

fchmod(2)

fchmodat(2) Added in POSIX.1-2008

fchown(2)

fchownat(2) Added in POSIX.1-2008

fcntl(2)

fdatasync(2)

fexecve(3) Added in POSIX.1-2008

ffs(3) Added in POSIX.1-2008 TC2

fork(2) See notes below

fstat(2)

fstatat(2) Added in POSIX.1-2008

fsync(2)

ftruncate(2)

futimens(3) Added in POSIX.1-2008

getegid(2)

geteuid(2)

getgid(2)

getgroups(2)

getpeername(2)

getpgrp(2)

getpid(2)

getppid(2)

getsockname(2)

getsockopt(2)

getuid(2)

htonl(3) Added in POSIX.1-2008 TC2

htons(3) Added in POSIX.1-2008 TC2

kill(2)

link(2)

linkat(2) Added in POSIX.1-2008

listen(2)

longjmp(3) Added in POSIX.1-2008 TC2; see notes below

lseek(2)

lstat(2)

memccpy(3) Added in POSIX.1-2008 TC2

memchr(3) Added in POSIX.1-2008 TC2

memcmp(3) Added in POSIX.1-2008 TC2

memcpy(3) Added in POSIX.1-2008 TC2

memmove(3) Added in POSIX.1-2008 TC2

memset(3) Added in POSIX.1-2008 TC2

mkdir(2)

mkdirat(2) Added in POSIX.1-2008

mkfifo(3)

mkfifoat(3) Added in POSIX.1-2008

mknod(2) Added in POSIX.1-2008

mknodat(2) Added in POSIX.1-2008

ntohl(3) Added in POSIX.1-2008 TC2

ntohs(3) Added in POSIX.1-2008 TC2

open(2)

openat(2) Added in POSIX.1-2008

pause(2)

pipe(2)

poll(2)

posix_trace_event(3)

pselect(2)

pthread_kill(3) Added in POSIX.1-2008 TC1

pthread_self(3) Added in POSIX.1-2008 TC1

pthread_sigmask(3) Added in POSIX.1-2008 TC1

raise(3)

read(2)

readlink(2)

readlinkat(2) Added in POSIX.1-2008

recv(2)

recvfrom(2)

recvmsg(2)

rename(2)

renameat(2) Added in POSIX.1-2008

rmdir(2)

select(2)

sem_post(3)

send(2)

sendmsg(2)

sendto(2)

setgid(2)

setpgid(2)

setsid(2)

setsockopt(2)

setuid(2)

shutdown(2)

sigaction(2)

sigaddset(3)

sigdelset(3)

sigemptyset(3)

sigfillset(3)

sigismember(3)

siglongjmp(3) Added in POSIX.1-2008 TC2; see notes below

signal(2)

sigpause(3)

sigpending(2)

sigprocmask(2)

sigqueue(2)

sigset(3)

sigsuspend(2)

sleep(3)

sockatmark(3) Added in POSIX.1-2001 TC2

socket(2)

socketpair(2)

stat(2)

stpcpy(3) Added in POSIX.1-2008 TC2

stpncpy(3) Added in POSIX.1-2008 TC2

strcat(3) Added in POSIX.1-2008 TC2

strchr(3) Added in POSIX.1-2008 TC2

strcmp(3) Added in POSIX.1-2008 TC2

strcpy(3) Added in POSIX.1-2008 TC2

strcspn(3) Added in POSIX.1-2008 TC2

strlen(3) Added in POSIX.1-2008 TC2

strncat(3) Added in POSIX.1-2008 TC2

strncmp(3) Added in POSIX.1-2008 TC2

strncpy(3) Added in POSIX.1-2008 TC2

strnlen(3) Added in POSIX.1-2008 TC2

strpbrk(3) Added in POSIX.1-2008 TC2

strrchr(3) Added in POSIX.1-2008 TC2

strspn(3) Added in POSIX.1-2008 TC2

strstr(3) Added in POSIX.1-2008 TC2

strtok_r(3) Added in POSIX.1-2008 TC2

symlink(2)

symlinkat(2) Added in POSIX.1-2008

tcdrain(3)

tcflow(3)

tcflush(3)

tcgetattr(3)

tcgetpgrp(3)

tcsendbreak(3)

tcsetattr(3)

tcsetpgrp(3)

time(2)

timer_getoverrun(2)

timer_gettime(2)

timer_settime(2)

times(2)

umask(2)

uname(2)

unlink(2)

unlinkat(2) Added in POSIX.1-2008

utime(2)

utimensat(2) Added in POSIX.1-2008

utimes(2) Added in POSIX.1-2008

wait(2)

waitpid(2)

wcpcpy(3) Added in POSIX.1-2008 TC2

wcpncpy(3) Added in POSIX.1-2008 TC2

wcscat(3) Added in POSIX.1-2008 TC2

wcschr(3) Added in POSIX.1-2008 TC2

wcscmp(3) Added in POSIX.1-2008 TC2

wcscpy(3) Added in POSIX.1-2008 TC2

wcscspn(3) Added in POSIX.1-2008 TC2

wcslen(3) Added in POSIX.1-2008 TC2

wcsncat(3) Added in POSIX.1-2008 TC2

wcsncmp(3) Added in POSIX.1-2008 TC2

wcsncpy(3) Added in POSIX.1-2008 TC2

wcsnlen(3) Added in POSIX.1-2008 TC2

wcspbrk(3) Added in POSIX.1-2008 TC2

wcsrchr(3) Added in POSIX.1-2008 TC2

wcsspn(3) Added in POSIX.1-2008 TC2

wcsstr(3) Added in POSIX.1-2008 TC2

wcstok(3) Added in POSIX.1-2008 TC2

wmemchr(3) Added in POSIX.1-2008 TC2

wmemcmp(3) Added in POSIX.1-2008 TC2

wmemcpy(3) Added in POSIX.1-2008 TC2

wmemmove(3) Added in POSIX.1-2008 TC2

wmemset(3) Added in POSIX.1-2008 TC2

write(2)

Notes:

* POSIX.1-2001 and POSIX.1-2001 TC2 required the functions fpathconf(3), pathconf(3), and sysconf(3) to be async-signal-safe, but this requirement was removed in POSIX.1-2008.

* If a signal handler interrupts the execution of an unsafe function, and the handler terminates via a call to longjmp(3) or siglongjmp(3) and the program subsequently calls an unsafe function, then the behavior of the program is undefined.

* POSIX.1-2001 TC1 clarified that if an application calls fork(2) from a signal handler and any of the fork handlers registered by pthread_atfork(3) calls a function that is not async-signal-safe, the behavior is undefined. A future revision of the standard is likely to remove fork(2) from the list of async-signal-safe functions.

* Asynchronous signal handlers that call functions which are cancellation points and nest over regions of deferred cancellation may trigger cancellation whose behavior is as if asynchronous cancellation had occurred and may cause application state to become incon? sistent.

errno

Fetching and setting the value of errno is async-signal-safe provided that the signal han? dler saves errno on entry and restores its value before returning.

Deviations in the GNU C library

The following known deviations from the standard occur in the GNU C library:

* Before glibc 2.24, execl(3) and execle(3) employed realloc(3) internally and were con? sequently not async-signal-safe. This was fixed in glibc 2.24.

* The glibc implementation of aio_suspend(3) is not async-signal-safe because it uses pthread_mutex_lock(3) internally.

SEE ALSO

sigaction(2), signal(7), standards(7)

COLOPHON

This page is part of release 5.10 of the Linux man-pages project. A description of the

project, information about reporting bugs, and the latest version of this page, can be found at https://www.kernel.org/doc/man-pages/.