



*Full credit is given to the above companies including the OS that this PDF file was generated!*

## ***Red Hat Enterprise Linux Release 9.2 Manual Pages on 'ulimit.3p' command***

**\$ man ulimit.3p**

ULIMIT(3P)            POSIX Programmer's Manual            ULIMIT(3P)

### PROLOG

This manual page is part of the POSIX Programmer's Manual. The Linux implementation of this interface may differ (consult the corresponding Linux manual page for details of Linux behavior), or the interface may not be implemented on Linux.

### NAME

ulimit ? get and set process limits

### SYNOPSIS

```
#include <ulimit.h>

long ulimit(int cmd, ...);
```

### DESCRIPTION

The `ulimit()` function shall control process limits. The process limits that can be controlled by this function include the maximum size of a single file that can be written (this is equivalent to using `setrlimit()` with `RLIMIT_FSIZE`). The `cmd` values, defined in `<ulimit.h>`, include:

`UL_GETFSIZE` Return the file size limit (`RLIMIT_FSIZE`) of the process.

The limit shall be in units of 512-byte blocks and shall be inherited by child processes. Files of any size can be read. The return value shall be the integer part of the soft file size limit divided by 512. If the result cannot be represented as a long, the result is unspecified.

UL\_SETFSIZE Set the file size limit for output operations of the process to the value of the second argument, taken as a long, multiplied by 512. If the result would overflow an rlim\_t, the actual value set is unspecified. Any process may decrease its own limit, but only a process with appropriate privileges may increase the limit. The return value shall be the integer part of the new file size limit divided by 512.

The ulimit() function shall not change the setting of errno if successful.

As all return values are permissible in a successful situation, an application wishing to check for error situations should set errno to 0, then call ulimit(), and, if it returns -1, check to see if errno is non-zero.

#### RETURN VALUE

Upon successful completion, ulimit() shall return the value of the requested limit. Otherwise, -1 shall be returned and errno set to indicate the error.

#### ERRORS

The ulimit() function shall fail and the limit shall be unchanged if:

EINVAL The cmd argument is not valid.

EPERM A process not having appropriate privileges attempts to increase its file size limit.

The following sections are informative.

#### EXAMPLES

None.

#### APPLICATION USAGE

Since the ulimit() function uses type long rather than rlim\_t, this function is not sufficient for file sizes on many current systems. Applications should use the getrlimit() or setrlimit() functions instead of the obsolescent ulimit() function.

#### RATIONALE

None.

## FUTURE DIRECTIONS

The `ulimit()` function may be removed in a future version.

## SEE ALSO

`exec`, `getrlimit()`, `write()`

The Base Definitions volume of POSIX.1-2017, `<ulimit.h>`

## COPYRIGHT

Portions of this text are reprinted and reproduced in electronic form from IEEE Std 1003.1-2017, Standard for Information Technology -- Portable Operating System Interface (POSIX), The Open Group Base Specifications Issue 7, 2018 Edition, Copyright (C) 2018 by the Institute of Electrical and Electronics Engineers, Inc and The Open Group. In the event of any discrepancy between this version and the original IEEE and The Open Group Standard, the original IEEE and The Open Group Standard is the referee document. The original Standard can be obtained online at <http://www.opengroup.org/unix/online.html> .

Any typographical or formatting errors that appear in this page are most likely to have been introduced during the conversion of the source files to man page format. To report such errors, see [https://www.kernel.org/doc/man-pages/reporting\\_bugs.html](https://www.kernel.org/doc/man-pages/reporting_bugs.html) .

IEEE/The Open Group

2017

ULIMIT(3P)