brk(2) — Linux manual page

NAME top

brk, sbrk - change data segment size

LIBRARY top

Standard C library (*libc*, *-lc*)

SYNOPSIS top

```
#include <unistd.h>

int brk(void *addr);
void *sbrk(intptr_t increment);
```

Feature Test Macro Requirements for glibc (see feature_test_macros(7)):

```
brk(), sbrk():
    Since glibc 2.19:
        _DEFAULT_SOURCE
            || ((_XOPEN_SOURCE >= 500) &&
                ! (_POSIX_C_SOURCE >= 200112L))
    From glibc 2.12 to glibc 2.19:
        _BSD_SOURCE || _SVID_SOURCE
            || ((_XOPEN_SOURCE >= 500) &&
                ! (_POSIX_C_SOURCE >= 200112L))
    Before glibc 2.12:
        _BSD_SOURCE || _SVID_SOURCE || _XOPEN_SOURCE >= 500
```

DESCRIPTION top

brk() and **sbrk**() change the location of the *program break*, which defines the end of the process's data segment (i.e., the program break is the first location after the end of the uninitialized data segment). Increasing the program break has the effect of allocating memory to the process; decreasing the break deallocates memory.

brk() sets the end of the data segment to the value specified by *addr*, when that value is reasonable, the system has enough memory, and the process does not exceed its maximum data size (see setrlimit(2)).

sbrk() increments the program's data space by *increment* bytes. Calling **sbrk**() with an *increment* of 0 can be used to find the current location of the program break.

RETURN VALUE top

On success, **brk**() returns zero. On error, -1 is returned, and *errno* is set to **ENOMEM**.

On success, **sbrk**() returns the previous program break. (If the break was increased, then this value is a pointer to the start of the newly allocated memory). On error, *(void *) -1* is returned, and *errno* is set to **ENOMEM**.

STANDARDS top

None.

HISTORY top

4.3BSD; SUSv1, marked LEGACY in SUSv2, removed in POSIX.1-2001.

NOTES top

Avoid using **brk**() and **sbrk**(): the malloc(3) memory allocation package is the portable and comfortable way of allocating memory.

Various systems use various types for the argument of **sbrk**(). Common are *int*, *ssize_t*, *ptrdiff_t*, *intptr_t*.

C library/kernel differences

The return value described above for **brk**() is the behavior provided by the glibc wrapper function for the Linux **brk**() system call. (On most other implementations, the return value from **brk**() is the same; this return value was also specified in SUSv2.) However, the actual Linux system call returns the new program break on success. On failure, the system call returns the current break. The glibc wrapper function does some work (i.e., checks whether the new break is less than *addr*) to provide the 0 and -1 return values described above.

On Linux, **sbrk**() is implemented as a library function that uses the **brk**() system call, and does some internal bookkeeping so that it can return the old break value.

SEE ALSO top

execve(2), getrlimit(2), end(3), malloc(3)

COLOPHON top

This page is part of the *man-pages* (Linux kernel and C library user-space interface documentation) project. Information about the project can be found at ⟨https://www.kernel.org/doc/man-pages/⟩. If you have a bug report for this manual page, see ⟨https://git.kernel.org/pub/scm/docs/man-pages/man-pages.git/tree/CONTRIBUTING⟩. This page was obtained from the tarball man-pages-6.10.tar.gz fetched from ⟨https://mirrors.edge.kernel.org/pub/linux/docs/man-pages/⟩ on 2025-02-02. If you discover any rendering problems in this HTML version of the page, or you believe there is a better or more up-to-date source for the page, or you have corrections or

```
       improvements to the information in this COLOPHON (which is not
       part of the original manual page), send a mail to
       man-pages@man7.org
```

Linux man-pages 6.10 **2024-07-23** ***brk*(2)**

Pages that refer to this page: getrlimit(2), mlock(2), mmap(2), mremap(2), PR_SET_MM_BRK(2const), PR_SET_MM_START_BRK(2const), PR_SET_TAGGED_ADDR_CTRL(2const), shmop(2), syscalls(2), alloca(3), end(3), malloc(3), malloc_hook(3), malloc_trim(3), mallopt(3), posix_memalign(3), proc_pid_stat(5), cpuset(7)
