



Linux Ubuntu 26.04 Manual Pages on command 'bzero.3'

\$ man bzero.3

bzero(3)
Library Functions Manual
bzero(3)

NAME

bzero, explicit bzero - zero a byte string

LIBRARY

Standard C library (libc, -lc)

SYNOPSIS

```
#include <strings.h>

void bzero(size_t n;
           void s[n], size_t n);

#include <string.h>

void explicit_bzero(size_t n;
                   void s[n], size_t n);
```

DESCRIPTION

The `bzero()` function erases the data in the `n` bytes of the memory starting at the location pointed to by `s`, by writing zeros (bytes containing `'\0'`) to that area.

The `explicit_bzero()` function performs the same task as `bzero()`. It differs from `bzero()` in that it guarantees that compiler optimizations will not remove the erase operation if the compiler deduces that the operation is "unnecessary".

RETURN VALUE

None.

ATTRIBUTES

For an explanation of the terms used in this section, see `attributes(7)`.

[illegible]

[illegible]

RAM that is then immediately erased (while the copy in the register remains unaffected). The problem here is that data in RAM is more likely to be exposed by a bug than data in a register, and thus the `explicit_bzero()` call creates a brief time window where the sensitive data is more vulnerable than it would otherwise have been if no attempt had been made to erase the data.

Note that declaring the sensitive variable with the `volatile` qualifier does not eliminate the above problems. Indeed, it will make them worse, since, for example, it may force a variable that would otherwise have been optimized into a register to instead be maintained in (more vulnerable) RAM for its entire lifetime.

Notwithstanding the above details, for security-conscious applications, using `explicit_bzero()` is generally preferable to not using it. The developers of `explicit_bzero()` anticipate that future compilers will recognize calls to `explicit_bzero()` and take steps to ensure that all copies of the sensitive data are erased, including copies in registers or in "scratch" stack areas.

SEE ALSO

`bstring(3)`, `memset(3)`, `swab(3)`

Linux man-pages 6.17

2026-02-08

`bzero(3)`