NAME

savecore - save a core dump of the operating system

SYNOPSIS

savecore -c [**-v**] [*device ...*]
savecore -C [**-v**] [*device ...*]
savecore -L [**-fvZz**] [**-m** *maxdumps*] [*directory*]
savecore [**--libxo**] [**-fkuvz**] [**-m** *maxdumps*] [*directory* [*device ...*]]

DESCRIPTION

The **savecore** utility copies a core dump into *directory*, or the current working directory if no *directory* argument is given, and enters a reboot message and information about the core dump into the system log.

The options are as follows:

--libxo Generate output via libxo(3) in a selection of different human and machine readable formats. See xo_parse_args(3) for details on command line arguments.

-C Check to see if a dump exists, and display a brief message to indicate the status. An exit status of 0 indicates that a dump is there, 1 indicates that none exists. This option is compatible only with the [**-v**] option.

-c Clear the dump, so that future invocations of **savecore** will ignore it.

-f Force a dump to be taken even if either the dump was cleared or if the dump header information is inconsistent.

-k Do not clear the dump after saving it.

-L Instruct **savecore** to generate and save a kernel dump of the running system, rather than copying one from a dump device.

-m *maxdumps* Maximum number of dumps to store. Once the number of stored dumps is equal to *maxdumps* the counter will restart from 0.

-u Uncompress the dump in case it was compressed by the kernel.

-v Print out some additional debugging information. Specify twice for more information.

-Z Compress the dump (see zstd(1)). This option is only supported in conjunction with the **-L** option. Regular dumps can be configured for compression with zstd using dumpon(8).

-z Compress the dump (see gzip(1)). The dump may already be compressed if the kernel was configured to do so by dumpon(8). In this case, the option has no effect.

If used in conjunction with the **-L** option, the requested live dump will be compressed with gzip.

The **savecore** utility looks for dumps on each device specified by the *device* argument(s), or on each device in */etc/fstab* marked as "dump" or "swap". The **savecore** utility checks the core dump in various ways to make sure that it is complete. If it passes these checks, it saves the core image in *directory/vmcore.#* and information about the core in *directory/info.#*. If the core is encrypted, it saves the dump key in *directory/key.#*. The core can be later decrypted using decryptcore(8). For kernel textdumps generated with the textdump(4) facility, output will be stored in the tar(5) format and named *directory/textdump.tar.#*. The "#" is the number from the first line of the file *directory/bounds*, and it is incremented and stored back into the file each time **savecore** successfully runs.

The **savecore** utility also checks the available disk space before attempting to make the copies. If there is insufficient disk space in the file system containing *directory*, or if the file *directory/minfree* exists and the number of free kilobytes (for non-superusers) in the file system after the copies were made would be less than the number in the first line of this file, the copies are not attempted.

If **savecore** successfully copies the kernel and the core dump, the core dump is cleared so that future invocations of **savecore** will ignore it.

The **savecore** utility is meant to be called near the end of the initialization file */etc/rc* (see rc(8)).

SEE ALSO

gzip(1), zstd(1), getbootfile(3), libxo(3), xo_parse_args(3), mem(4), textdump(4), tar(5), crashinfo(8), decryptcore(8), dumpon(8), syslogd(8)

HISTORY

The **savecore** utility appeared in 4.1BSD.

Support for kernel textdumps appeared in FreeBSD 7.1.

BUGS

The minfree code does not consider the effect of compression or sparse files.