NAME

scrub − write patterns on disk/file

SYNOPSIS

scrub *[OPTIONS] special-file [special-file ...]*
scrub *[OPTIONS] file [file ...]*
scrub -*X [OPTIONS] directory*

DESCRIPTION

Scrub iteratively writes patterns on files or disk devices to make retrieving the data more difficult. **Scrub** operates in one of three modes:

1) The special file corresponding to an entire disk is scrubbed and all data on it is destroyed. This mode is selected if *file* is a character or block special file. This is the most ef fective method.

2) A regular file is scrubbed and only the data in the file (and optionally its name in the directory entry) is destroyed. The file size is rounded up to fill out the last file system block. This mode is selected if*file* is a regular file. See CAVEATS below.

3) *directory* is created and filled with files until the file system is full, then the files are scrubbed as in 2). This mode is selected with the -*X* option. See CAVEATS below.

OPTIONS

Scrub accepts the following options:

-*v*, --*version*
: Print scrub version and exit.

-*r*, --*remove*
: Remove the file after scrubbing.

-*p*, --*pattern PATTERN*
: Select the patterns to write. See SCRUB METHODS below. The default, *nnsa*, is reasonable for sanitizing modern PRML/EPRML encoded disk devices.

-*b*, --*blocksize blocksize*
: Perform read(2) and write(2) calls using the specified blocksize (in bytes). *K*, *M*, or *G* may be appended to the number to change the units to KiBytes, MiBytes, or GiBytes, respectively. Default: 4M.

-*f*, --*force*
: Scrub even if target contains signature indicating it has already been scrubbed.

-*S*, --*no-signature*
: Do not write scrub signature. Later, **scrub** will not be able to ascertain if the disk has already been scrubbed.

-*X*, --*freespace*
: Create specified directory and fill it with files until write returns ENOSPC (file system full), then scrub the files as usual. The size of each file can be set with -*s*, otherwise it will be the maximum file size creatable given the user's file size limit or 1g if unlimited.

-*D*, --*dirent newname*
: After scrubbing the file, scrub its name in the directory entry, then rename it to the new name. The scrub patterns used on the directory entry are constrained by the operating system and thus are not compliant with cited standards. This option only works with a single target.

-*s*, --*device-size size*
: Override the device size (in bytes). Without this option, **scrub** determines media capacity using OS-specific ioctl(2) calls. *K*, *M*, or *G* may be appended to the number to change the units to KiBytes, MiBytes, or GiBytes, respectively.

-L, --no-link
If *file* is a symbolic link, do not scrub the link target. Do remove it, however, if *--remove* is specified.

-R, --no-hwrand
Don't use a hardware random number generator even if one is available.

-t, --no-threads
Don't generate random data in parallel with I/O.

-n, --dry-run
Do everything but write to targets.

-h, --help
Print a summary of command line options on stderr.

SCRUB METHODS

nnsa 4-pass NNSA Policy Letter NAP-14.1-C (XVI-8) for sanitizing removable and non-removable hard disks, which requires overwriting all locations with a pseudorandom pattern twice and then with a known pattern: **random(x2), 0x00, verify**.

dod 4-pass DoD 5220.22-M section 8-306 procedure (d) for sanitizing removable and non-removable rigid disks which requires overwriting all addressable locations with a character, its complement, a random character, then verify. NOTE: **scrub** performs the random pass first to make verification easier: **random, 0x00, 0xff, verify**.

bsi 9-pass method recommended by the German Center of Security in Information Technologies (http://www.bsi.bund.de): **0xff, 0xfe, 0xfd, 0xfb, 0xf7, 0xef, 0xdf, 0xbf, 0x7f**.

gutmann
The canonical 35-pass sequence described in Gutmann's paper cited below.

schneier
7-pass method described by Bruce Schneier in "Applied Cryptography" (1996): **0x00, 0xff, random(x5)**

pfitzner7
Roy Pfitzner's 7-random-pass method: **random(x7)**.

pfitzner33
Roy Pfitzner's 33-random-pass method: **random(x33)**.

usarmy US Army AR380-19 method: **0x00, 0xff, random**. (Note: identical to DoD 522.22-M section 8-306 procedure (e) for sanitizing magnetic core memory).

fillzero 1-pass pattern: **0x00**.

fillff 1-pass pattern: **0xff**.

random
1-pass pattern: **random(x1)**.

random2
2-pass pattern: **random(x2)**.

old 6-pass pre-version 1.7 scrub method: **0x00, 0xff, 0xaa, 0x00, 0x55, verify**.

fastold 5-pass pattern: **0x00, 0xff, 0xaa, 0x55, verify**.

custom=string
1-pass custom pattern. String may contain C-style numerical escapes: \nnn (octal) or \xnn (hex).

CAVEATS

Scrub may be insufficient to thwart heroic efforts to recover data in an appropriately equipped lab. If you need this level of protection, physical destruction is your best bet.

The effectiveness of scrubbing regular files through a file system will be limited by the OS and file system.

File systems that are known to be problematic are journaled, log structured, copy-on-write, versioned, and network file systems. If in doubt, scrub the raw disk device.

Scrubbing free blocks in a file system with the *-X* method is subject to the same caveats as scrubbing regular files, and in addition, is only useful to the extent the file system allows you to reallocate the target blocks as data blocks in a new file. If in doubt, scrub the raw disk device.

On MacOS X HFS file system, **scrub** attempts to overwrite a file's resource fork if it exists. Although MacOS X claims it will support additional named forks in the future, **scrub** is only aware of the traditional data and resource forks.

scrub cannot access disk blocks that have been spared out by the disk controller. For SATA/PATA drives, the ATA "security erase" command built into the drive controller can do this. Similarly, the ATA "enhanced security erase" can erase data on track edges and between tracks. The DOS utility HDDERASE from the UCSD Center for Magnetic Recording Research can issue these commands, as can modern versions of Linux **hdparm**. Unfortunately, the analogous SCSI command is optional according to T-10, and not widely implemented.

EXAMPLES

To scrub a raw device */dev/sdf1* with default NNSA patterns:

```
# scrub /dev/sdf1
scrub: using NNSA NAP-14.1-C patterns
scrub: please verify that device size below is correct!
scrub: scrubbing /dev/sdf1 1995650048 bytes (~1GB)
scrub: random  |.....................................|
scrub: random  |.....................................|
scrub: 0x00    |.....................................|
scrub: verify  |.....................................|
```

To scrub the file */tmp/scrubme* with a sequence of 0xff 0xaa bytes:

```
# scrub -p custom="\xff\xaa" /tmp/scrubme
scrub: using Custom single-pass patterns
scrub: scrubbing /tmp/scrubme 78319616 bytes (~74MB)
scrub: 0xffaa  |.....................................|
```

AUTHOR

Jim Garlick <garlick@llnl.gov>

SEE ALSO

DoD 5220.22-M, "National Industrial Security Program Operating Manual", Chapter 8, 01/1995.

NNSA Policy Letter: NAP-14.1-C, "Clearing, Sanitizing, and Destroying Information System Storage Media, Memory Devices, and other Related Hardware", 05-02-08, page XVI-8.

"Secure Deletion of Data from Magnetic and Solid-State Memory", by Peter Gutmann, Sixth USENIX Security Symposium, San Jose, CA, July 22-25, 1996.

"Gutmann Method", Wikipedia, http://en.wikipedia.org/wiki/Gutmann_method.

Darik's boot and Nuke FAQ: http://dban.sourceforge.net/faq/index.html

"Tutorial on Disk Drive Data Sanitization", by Gordon Hugues and Tom Coughlin, http://cmrr.ucsd.edu/people/Hughes/DataSanitizationTutorial.pdf.

"Guidelines for Media Sanitization", NIST special publication 800-88, Kissel et al, September, 2006.

shred(1), hdparm(8)