

USING FSCK

Mark E. Donaldson

fsck is a Unix utility for checking and repairing file system inconsistencies. File system can become inconsistent due to several reasons and the most common is abnormal shutdown due to hardware failure, power failure or switching off the system without proper shutdown. Due to these reasons the superblock in a file system is not updated and has mismatched information relating to system data blocks, free blocks and inodes.

Modes of operation:

fsck operates in two modes interactive and non interactive :

interactive: the fsck examines the file system and stops at each error it finds in the file system and gives the problem description and ask for user response usually whether to correct the problem or continue without making any change to the file system.

noninteractive: fsck tries to repair all the problems it finds in a file system without stopping for user response useful in case of a large number of inconsistencies in a file system but has the disadvantage of removing some useful files which are detected to be corrupt .

If file system is found to have problem at the booting time non interactive fsck fsck is run and all errors which are considered safe to correct are corrected. But if still file system has problems the system boots in single user mode asking for user to manually run the fsck to correct the problems in file system.

Running fsck

fsck should always be run in a single user mode which ensures proper repair of file system . If it is run in a busy system where the file system is changing constantly fsck may see the changes as inconsistencies and may corrupt the file system .

if the system can not be brought in a single user mode fsck should be run on the partitions, other than root and usr, after unmounting them . Root and usr partitions can not be unmounted. If the system fails to come up due to root/usr files system corruption the system can booted with CD and root/usr partitions can be repaired using fsck.

Command Syntax

fsck [**-F** *fstype*] [**-V**] [**-yY**] [**-o** *options*] *special*

-F *fstype* type of file system to be repaired (ufs , vxfs etc)

-V verify the command line syntax but do not run the command

-y or **-Y** Run the command in non interactive mode - repair all errors encountered without waiting for user response.

-o *options* Three options can be specified with -o flag

b=n where n is the number of next super block if primary super block is corrupted in a file system .

p option used to make safe repair options during the booting process.

f force the file system check regardless of its clean flag.

USING FSCK

Mark E. Donaldson

special - Block or character device name of the file system to be checked/repared - for example /dev/rdisk/c0t3d0s4 .Character device should be used for consistencies check & repair

Phases

fsck checks the file system in a series of 5 pages and checks a specific functionality of file system in each phase.

1. phase 1 - Check Blocks and Sizes
2. phase 2 - Check Pathnames
3. phase 3 - Check Connectivity
4. phase 4 - Check Reference Counts
5. phase 5 - Check Cylinder Groups

Error messages & Corrective Action

1. **Corrupted superblock** - fsck fails to run

If the superblock is corrupted the file system still can be repaired using alternate superblock which are formed while making new file system.

the first alternate superblock number is 32 and others superblock numbers can be found using the following command :

```
newfs -N /dev/rdisk/c0t0d0s6
```

for example to run fsck using first alternate superblock following command is used

```
fsck -F ufs -o b=32 /dev/rdisk/c0t0d0s6
```

2. **Link counter adjustment** - fsck finds mismatch between directory inode link counts and actual directory links and prompts for adjustment in case of interactive operation .Link count adjustments are considered to be a safe operation in a file system and should be repaired by giving 'y' response to the adjust ? prompt during fsck.
3. **Free Block count salvage** - During fsck the number of free blocks listed in a superblock and actual unallocated free blocks count does not match .fsck inform this mismatch and asks to salvage free block count to synchronize the superblock count. This error can be corrected without any potential problem to the file system or files.
4. **Unreferenced file reconnection** - While checking connectivity, fsck finds some inodes which are allocated but not referenced -not attached to any directory. Answering y to reconnect message by fsck links these files to the lost+found directory with their inode number as their name.

To get more info about the files in lost+found 'file' command can be used to see the type of files and subsequently they can be opened in their applications or text editors to find out about their contents. If the file is found to be correct it can be used after copying to some other directory and renaming it.

USING FSCK

Mark E. Donaldson

If you want to check or repair your linux filesystem you need to use fsck command. The native Linux filesystem (ext2) does not need to be defragmented. However, occasionally, you may need to check a partition's file allocation and make repairs.

Type: fsck options filesystem or fsck options mountpoint .

By default, repairs are done without prompting.

Two ways of specifying a filesystem exist. You do not need to specify a filesystem if you use the -t or -A options. This check is also run when the system starts if it was not shut down gracefully or after a predefined number of reboots.

fsck manpage

NAME_

fsck - check and repair a Linux file system

SYNOPSIS_

```
fsck [ -sAVRTNP ] [ -C [ fd ] ] [ -t fstype ] [filesys ... ] [--] [ fs-specific-options ]
```

DESCRIPTION_

fsck is used to check and optionally repair one or more Linux file systems. *filesys* can be a device name (e.g. */dev/hdc1*, */dev/sdb2*), a mount point (e.g. */*, */usr*, */home*), or an ext2 label or UUID specifier (e.g. *UUID=8868abf6-88c5-4a83-98b8-bfc24057f7bd* or *LABEL=root*). Normally, the **fsck** program will try to handle filesystems on different physical disk drives in parallel to reduce the total amount of time needed to check all of the filesystems.

If no filesystems are specified on the command line, and the **-A** option is not specified, **fsck** will default to checking filesystems in **/etc/fstab** serially. This is equivalent to the **-As** options.

The exit code returned by **fsck** is the sum of the following conditions: 0 - No errors 1 - File system errors corrected 2 - System should be rebooted 4 - File system errors left uncorrected 8 - Operational error 16 - Usage or syntax error 32 - Fsck canceled by user request 128 - Shared library error The exit code returned when multiple file systems are checked is the bit-wise OR of the exit codes for each file system that is checked.

In actuality, **fsck** is simply a front-end for the various file system checkers (**fsck.fstype**) available under Linux. The file system-specific checker is searched for in */sbin* first, then in */etc/fs* and */etc*, and finally in the directories listed in the *PATH* environment variable. Please see the file system-specific checker manual pages for further details.

USING FSCK

Mark E. Donaldson

OPTIONS_

-s

Serialize **fsck** operations. This is a good idea if you are checking multiple filesystems and the checkers are in an interactive mode. (Note: **e2fsck** runs in an interactive mode by default. To make **e2fsck** run in a non-interactive mode, you must either specify the **-p** or **-a** option, if you wish for errors to be corrected automatically, or the **-n** option if you do not.)

-t *fslist*

Specifies the type(s) of file system to be checked. When the **-A** flag is specified, only filesystems that match *fslist* are checked. The *fslist* parameter is a comma-separated list of filesystems and options specifiers. All of the filesystems in this comma-separated list may be prefixed by a negation operator '**no**' or '**!**', which requests that only those filesystems not listed in *fslist* will be checked. If all of the filesystems in *fslist* are not prefixed by a negation operator, then only those filesystems listed in *fslist* will be checked.

Options specifiers may be included in the comma-separated *fslist*. They must have the format **opts=fs-option**. If an options specifier is present, then only filesystems which contain *fs-option* in their mount options field of **/etc/fstab** will be checked. If the options specifier is prefixed by a negation operator, then only those filesystems that do not have *fs-option* in their mount options field of **/etc/fstab** will be checked.

For example, if **opts=ro** appears in *fslist*, then only filesystems listed in **/etc/fstab** with the **ro** option will be checked.

For compatibility with Mandrake distributions whose boot scripts depend upon an unauthorized UI change to the **fsck** program, if a filesystem type of **loop** is found in *fslist*, it is treated as if **opts=loop** were specified as an argument to the **-t** option.

Normally, the filesystem type is deduced by searching for *filesys* in the **/etc/fstab** file and using the corresponding entry. If the type can not be deduced, and there is only a single filesystem given as an argument to the **-t** option, **fsck** will use the specified filesystem type. If this type is not available, then the default file system type (currently ext2) is used.

-A

Walk through the **/etc/fstab** file and try to check all file systems in one run. This option is typically used from the **/etc/rc** system initialization file, instead of multiple commands for checking a single file system.

The root filesystem will be checked first unless the **-P** option is specified (see below). After that, filesystems will be checked in the order specified by the *fs_passno* (the sixth) field in the **/etc/fstab** file. Filesystems with a *fs_passno* value of 0 are skipped and are not checked at all. Filesystems with a *fs_passno* value of greater than zero will be checked in order, with filesystems with the lowest *fs_passno* number being checked first. If there are multiple filesystems with the same pass number, **fsck** will attempt to check them in parallel, although it will avoid running multiple filesystem checks on the same physical disk.

USING FSCK

Mark E. Donaldson

Hence, a very common configuration in */etc/fstab* files is to set the root filesystem to have a *fs_passno* value of 1 and to set all other filesystems to have a *fs_passno* value of 2. This will allow **fsck** to automatically run filesystem checkers in parallel if it is advantageous to do so. System administrators might choose not to use this configuration if they need to avoid multiple filesystem checks running in parallel for some reason --- for example, if the machine in question is short on memory so that excessive paging is a concern.

-C [*fd*]

Display completion/progress bars for those filesystem checkers (currently only for ext2 and ext3) which support them. Fscck will manage the filesystem checkers so that only one of them will display a progress bar at a time. GUI front-ends may specify a file descriptor *fd*, in which case the progress bar information will be sent to that file descriptor.

-N

Don't execute, just show what would be done.

-P

When the **-A** flag is set, check the root filesystem in parallel with the other filesystems. This is not the safest thing in the world to do, since if the root filesystem is in doubt things like the **e2fsck** executable might be corrupted! This option is mainly provided for those sysadmins who don't want to repartition the root filesystem to be small and compact (which is really the right solution).

-R

When checking all file systems with the **-A** flag, skip the root file system (in case it's already mounted read-write).

-T

Don't show the title on startup.

-V

Produce verbose output, including all file system-specific commands that are executed.

fs-specific-options

Options which are not understood by **fsck** are passed to the filesystem-specific checker. These arguments **must** not take arguments, as there is no way for **fsck** to be able to properly guess which arguments take options and which don't.

Options and arguments which follow the **--** are treated as file system-specific options to be passed to the file system-specific checker.

Please note that **fsck** is not designed to pass arbitrarily complicated options to filesystem-specific checkers. If you're doing something complicated, please just execute the filesystem-specific checker directly. If you pass **fsck** some horribly complicated option and arguments, and it doesn't do what you expect, **don't bother reporting it as a bug**. You're almost certainly doing something that you shouldn't be doing with **fsck**.

Options to different filesystem-specific **fsck**'s are not standardized. If in doubt, please consult the man pages of the filesystem-specific checker. Although not guaranteed, the following options are supported by most file system checkers:

-a

USING FSCK

Mark E. Donaldson

Automatically repair the file system without any questions (use this option with caution). Note that **e2fsck** supports **-a** for backwards compatibility only. This option is mapped to **e2fsck**'s **-p** option which is safe to use, unlike the **-a** option that some file system checkers support.

-n

For some filesystem-specific checkers, the **-n** option will cause the fs-specific fsck to avoid attempting to repair any problems, but simply report such problems to stdout. This is however not true for all filesystem-specific checkers. In particular, **fsck.reiserfs(8)** will not report any corruption if given this option. **fsck.minix(8)** does not support the **-n** option at all.

-r

Interactively repair the filesystem (ask for confirmations). Note: It is generally a bad idea to use this option if multiple fsck's are being run in parallel. Also note that this is **e2fsck**'s default behavior; it supports this option for backwards compatibility reasons only.

-y

For some filesystem-specific checkers, the **-y** option will cause the fs-specific fsck to always attempt to fix any detected filesystem corruption automatically. Sometimes an expert may be able to do better driving the fsck manually. Note that **not** all filesystem-specific checkers implement this option. In particular **fsck.minix** and **fsck.cramfs(8)** does not support the **-y** option as of this writing.

FILES_

/etc/fstab.

ENVIRONMENT VARIABLES_

The **fsck** program's behavior is affected by the following environment variables:

FSCK_FORCE_ALL_PARALLEL

If this environment variable is set, **fsck** will attempt to run all of the specified filesystems in parallel, regardless of whether the filesystems appear to be on the same device. (This is useful for RAID systems or high-end storage systems such as those sold by companies such as IBM or EMC.)

FSCK_MAX_INST

This environment variable will limit the maximum number of file system checkers that can be running at one time. This allows configurations which have a large number of disks to avoid **fsck** starting too many file system checkers at once, which might overload CPU and memory resources available on the system. If this value is zero, then an unlimited number of processes can be spawned. This is currently the default, but future versions of **fsck** may attempt to automatically determine how many file system checks can be run based on gathering accounting data from the operating system.

PATH

The **PATH** environment variable is used to find file system checkers. A set of system directories are searched first: **/sbin**, **/sbin/fs.d**, **/sbin/fs**, **/etc/fs**, and **/etc**. Then the set of directories found in the **PATH** environment are searched.

FSTAB_FILE

This environment variable allows the system administrator to override the standard location of the **/etc/fstab** file. It is also useful for developers who are testing **fsck**.