Character pointers are dereferenced and printed as C strings. Non-printing characters in strings are normally represented by ordinary C escape codes. Only the first *strsize* (32 by default) bytes of strings are printed; longer strings have an ellipsis appended following the closing quote. Here is a line from ''ls -l'' where the **getpwuid** library routine is reading the password file:

```
read(3, "root::0:0:System Administrator:/"..., 1024) = 422
```

While structures are annotated using curly braces, simple pointers and arrays are printed using square brackets with commas separating elements. Here is an example from the command ''id'' on a system with supplementary group ids:

```
getgroups(32, [100, 0]) = 2
```

On the other hand, bit-sets are also shown using square brackets but set elements are separated only by a space. Here is the shell preparing to execute an external command:

```
sigprocmask(SIG_BLOCK, [CHLD TTOU], []) = 0
```

Here the second argument is a bit-set of two signals, SIGCHLD and SIGTTOU. In some cases the bit-set is so full that printing out the unset elements is more valuable. In that case, the bit-set is prefixed by a tilde like this:

```
sigprocmask(SIG_UNBLOCK, ~[], NULL) = 0
```

Here the second argument represents the full set of all signals.

Options

-c

Count time, calls, and errors for each system call and report a summary on program exit. On Linux, this attempts to show system time (CPU time spent running in the kernel) independent of wall clock time. If -c is used with -f or -F (below), only aggregate totals for all traced processes are kept.

-D

(Not available on SVR4 and FreeBSD.) Run tracer process as a detached grandchild, not as parent of the tracee. This reduces the visible effect of **strace** by keeping the tracee a direct child of the calling process.

-d

Show some debugging output of **strace** itself on the standard error.

-f

Trace child processes as they are created by currently traced processes as a result of the **fork**(2) system call.

> On non-Linux platforms the new process is attached to as soon as its pid is known (through the return value of **fork**(2) in the parent process). This means that such children may run uncontrolled for a while (especially in the case of a **vfork**(2)), until the parent is scheduled again to complete its (**v**)**fork**(2) call. On Linux the child is traced from its first instruction with no delay. If the parent process decides to **wait**(2) for a child that is currently being traced, it is suspended until an appropriate child process either terminates or incurs a signal that would cause it to terminate (as determined from the child's current signal disposition).
>
> On SunOS 4.x the tracing of **vfork**s is accomplished with some dynamic linking trickery.

-ff

If the **-o** *filename* option is in effect, each processes trace is written to *filename.pid* where pid is the numeric process id of each process. This is incompatible with -c, since no per-process counts are kept.

-F

This option is now obsolete and it has the same functionality as **-f**.

-h

Print the help summary.

-i

Print the instruction pointer at the time of the system call.

-q

Suppress messages about attaching, detaching etc. This happens automatically when output is redirected to a file and the command is run directly instead of attaching.

-r

Print a relative timestamp upon entry to each system call. This records the time difference between the beginning of successive system calls.

-t

Prefix each line of the trace with the time of day.

-tt

If given twice, the time printed will include the microseconds.

-ttt

If given thrice, the time printed will include the microseconds and the leading portion will be printed as the number of seconds since the epoch.

-T

Show the time spent in system calls. This records the time difference between the beginning and the end of each system call.

-v

Print unabbreviated versions of environment, stat, termios, etc. calls. These structures are very common in calls and so the default behavior displays a reasonable subset of structure members. Use this option to get all of the gory details.

-V

Print the version number of **strace**.

-x

Print all non-ASCII strings in hexadecimal string format.

-xx

Print all strings in hexadecimal string format.

-a *column*

Align return values in a specific column (default column 40).

-e *expr*

A qualifying expression which modifies which events to trace or how to trace them. The format of the expression is:

[*qualifier***=**][**!**]*value1*[**,***value2*]...

> where *qualifier* is one of **trace**, **abbrev**, **verbose**, **raw**, **signal**, **read**, or **write** and *value* is a qualifier-dependent symbol or number. The default qualifier is **trace**. Using an exclamation mark negates the set of values. For example, **-eopen** means literally **-e trace=open** which in turn means trace only the **open** system call. By contrast, **-etrace=!open** means to trace every system call except **open**. In addition, the special values **all** and **none** have the obvious meanings.

Note that some shells use the exclamation point for history expansion even inside quoted arguments. If so, you must escape the exclamation point with a backslash.

-e trace=*set*
Trace only the specified set of system calls. The **-c** option is useful for determining which system calls might be useful to trace. For example, **trace=open,close,read,write** means to only trace those four system calls. Be careful when making inferences about the user/kernel boundary if only a subset of system calls are being monitored. The default is **trace=all**.

-e trace=file
Trace all system calls which take a file name as an argument. You can think of this as an abbreviation for **-e trace=open,stat,chmod,unlink,**... which is useful to seeing what files the process is referencing. Furthermore, using the abbreviation will ensure that you don't accidentally forget to include a call like **lstat** in the list. Betchya woulda forgot that one.

-e trace=process
Trace all system calls which involve process management. This is useful for watching the fork, wait, and exec steps of a process.

-e trace=network
Trace all the network related system calls.

-e trace=signal
Trace all signal related system calls.

-e trace=ipc
Trace all IPC related system calls.

-e trace=desc
Trace all file descriptor related system calls.

-e abbrev=*set*
Abbreviate the output from printing each member of large structures. The default is **abbrev=all**. The **-v** option has the effect of **abbrev=none**.

-e verbose=*set*
Dereference structures for the specified set of system calls. The default is **verbose=all**.

-e raw=*set*

Print raw, undecoded arguments for the specified set of system calls. This option has the effect of causing all arguments to be printed in hexadecimal. This is mostly useful if you don't trust the decoding or you need to know the actual numeric value of an argument.

-e signal=*set*
Trace only the specified subset of signals. The default is **signal=all**. For example, **signal=!SIGIO** (or **signal=!io**) causes SIGIO signals not to be traced.

-e read=*set*

Perform a full hexadecimal and ASCII dump of all the data read from file descriptors listed in the specified set. For example, to see all input activity on file descriptors 3 and 5 use **-e**

read=3,5. Note that this is independent from the normal tracing of the **read**(2) system call which is controlled by the option **-e trace=read**.

-e write=*set*

Perform a full hexadecimal and ASCII dump of all the data written to file descriptors listed in the specified set. For example, to see all output activity on file descriptors 3 and 5 use **-e write=3,5**. Note that this is independent from the normal tracing of the **write**(2) system call which is controlled by the option **-e trace=write**.

-o *filename*

Write the trace output to the file *filename* rather than to stderr. Use *filename.pid* if **-ff** is used. If the argument begins with '|' or with '!' then the rest of the argument is treated as a command and all output is piped to it. This is convenient for piping the debugging output to a program without affecting the redirections of executed programs.

-O *overhead*

Set the overhead for tracing system calls to *overhead* microseconds. This is useful for overriding the default heuristic for guessing how much time is spent in mere measuring when timing system calls using the **-c** option. The accuracy of the heuristic can be gauged by timing a given program run without tracing (using **time**(1)) and comparing the accumulated system call time to the total produced using **-c**.

-p *pid*

Attach to the process with the process ID *pid* and begin tracing. The trace may be terminated at any time by a keyboard interrupt signal (CTRL -C). **strace** will respond by detaching itself from the traced process(es) leaving it (them) to continue running. Multiple **-p** options can be used to attach to up to 32 processes in addition to *command* (which is optional if at least one **-p** option is given).

-s *strsize*

Specify the maximum string size to print (the default is 32). Note that filenames are not considered strings and are always printed in full.

-S *sortby*

Sort the output of the histogram printed by the **-c** option by the specified criterion. Legal values are **time**, **calls**, **name**, and **nothing** (default **time**).

-u *username*

Run command with the user ID , group ID , and supplementary groups of *username*. This option is only useful when running as root and enables the correct execution of setuid and/or setgid binaries. Unless this option is used setuid and setgid programs are executed without effective privileges.

-E *var=val*

Run command with *var=val* in its list of environment variables.

-E *var*

Remove *var* from the inherited list of environment variables before passing it on to the command.

Diagnostics

When *command* exits, **strace** exits with the same exit status. If *command* is terminated by a signal, **strace** terminates itself with the same signal, so that **strace** can be used as a wrapper process transparent to the invoking parent process.

When using **-p**, the exit status of **strace** is zero unless there was an unexpected error in doing the tracing.

Setuid Installation

If **strace** is installed setuid to root then the invoking user will be able to attach to and trace processes owned by any user. In addition setuid and setgid programs will be executed and traced with the correct effective privileges. Since only users trusted with full root privileges should be allowed to do these things, it only makes sense to install **strace** as setuid to root when the users who can execute it are restricted to those users who have this trust. For example, it makes sense to install a special version of **strace** with mode 'rwsr-xr--', user **root** and group **trace**, where members of the **trace** group are trusted users. If you do use this feature, please remember to install a non-setuid version of **strace** for ordinary lusers to use.

See Also

ltrace(1), **time**(1), **ptrace**(2), **proc**(5)

Notes

It is a pity that so much tracing clutter is produced by systems employing shared libraries.

It is instructive to think about system call inputs and outputs as data-flow across the user/kernel boundary. Because user-space and kernel-space are separate and address-protected, it is sometimes possible to make deductive inferences about process behavior using inputs and outputs as propositions.

In some cases, a system call will differ from the documented behavior or have a different name. For example, on System V-derived systems the true **time**(2) system call does not take an argument and the **stat** function is called **xstat** and takes an extra leading

od(1) - Linux man page

Name

od - dump files in octal and other formats

Synopsis

od [*OPTION*]... [*FILE*]...
od [*-abcdfilosx*]... [*FILE*] [[+]*OFFSET*[*.*][*b*]]
od *--traditional* [*OPTION*]... [*FILE*] [[+]*OFFSET*[*.*][*b*] [+][*LABEL*][*.*][*b*]]

Description

Write an unambiguous representation, octal bytes by default, of FILE to standard output. With more than one FILE argument, concatenate them in the listed order to form the input. With no FILE, or when FILE is -, read standard input.

All arguments to long options are mandatory for short options.

-A, **--address-radix**=*RADIX*
decide how file offsets are printed
-j, **--skip-bytes**=*BYTES*
skip BYTES input bytes first
-N, **--read-bytes**=*BYTES*
limit dump to BYTES input bytes
-S, **--strings**[=*BYTES*]
output strings of at least BYTES graphic chars
-t, **--format**=*TYPE*
select output format or formats
-v, **--output-duplicates**
do not use * to mark line suppression
-w, **--width**[=*BYTES*]
output BYTES bytes per output line
--traditional
accept arguments in traditional form
--help
display this help and exit
--version
output version information and exit

Traditional format specifications may be intermixed; they accumulate:

-a
same as **-t** a, select named characters, ignoring high-order bit

-b
same as **-t** o1, select octal bytes

-c
same as **-t** c, select ASCII characters or backslash escapes

-d
same as **-t** u2, select unsigned decimal 2-byte units

-f
same as **-t** fF, select floats

-i
same as **-t** dI, select decimal ints

-l
same as **-t** dL, select decimal longs

-o
same as **-t** o2, select octal 2-byte units

-s
same as **-t** d2, select decimal 2-byte units

-x
same as **-t** x2, select hexadecimal 2-byte units

If first and second call formats both apply, the second format is assumed if the last operand begins with + or (if there are 2 operands) a digit. An OFFSET operand means **-j** OFFSET. LABEL is the pseudo-address at first byte printed, incremented when dump is progressing. For OFFSET and LABEL, a 0x or 0X prefix indicates hexadecimal; suffixes may be . for octal and b for multiply by 512.

TYPE is made up of one or more of these specifications:

a
named character, ignoring high-order bit

c
ASCII character or backslash escape

d[SIZE]
signed decimal, SIZE bytes per integer

f[SIZE]
floating point, SIZE bytes per integer

o[SIZE]
octal, SIZE bytes per integer

u[SIZE]
unsigned decimal, SIZE bytes per integer

x[SIZE]
hexadecimal, SIZE bytes per integer

SIZE is a number. For TYPE in doux, SIZE may also be C for sizeof(char), S for sizeof(short), I for sizeof(int) or L for *sizeof(long)*. If TYPE is f, SIZE may also be F for sizeof(float), D for sizeof(double) or L for sizeof(long double).

kill(1) - Linux man page

Name

kill - terminate a process

Synopsis

kill [**-s** *signal*|**-p**] [**--**] *pid*...
kill -l [*signal*]

Description

The command **kill** sends the specified signal to the specified process or process group. If no signal is specified, the TERM signal is sent. The TERM signal will kill processes which do not catch this signal. For other processes, it may be necessary to use the KILL (9) signal, since this signal cannot be caught.

Most modern shells have a builtin kill function, with a usage rather similar to that of the command described here. The '-a' and '-p' options, and the possibility to specify pids by command name is a local extension.

If sig is 0, then no signal is sent, but error checking is still performed.

Options

pid...

Specify the list of processes that **kill** should signal. Each *pid* can be one of five things:

n

where *n* is larger than 0. The process with pid *n* will be signaled.

0

All processes in the current process group are signaled.

-1

All processes with pid larger than 1 will be signaled.

-*n*

where *n* is larger than 1. All processes in process group *n* are signaled. When an argument of the form '-n' is given, and it is meant to denote a process group, either the signal must be specified first, or the argument must be preceded by a '--' option, otherwise it will be taken as the signal to send.

commandname

All processes invoked using that name will be signaled.

-s *signal*
Specify the signal to send. The signal may be given as a signal name or number.
-l

Print a list of signal names. These are found in */usr/include/linux/signal.h*

-a

Do not restrict the commandname-to-pid conversion to processes with the same uid as the present process.

-p

Specify that **kill** should only print the process id (pid) of the named processes, and not send any signals.

See Also

bash(1), **tcsh**(1), **kill**(2), **sigvec**(2), **signal**(7)

Author

Taken from BSD 4.4. The ability to translate process names to process ids was added by Salvatore Valente <svalente@mit.edu>.

Availability

The kill command is part of the util-linux-ng package and is available from ftp://ftp.kernel.org/pub/linux/utils/util-linux-ng/.

Referenced By

3proxy(8), **darkstat**(8), **forktest**(6), **fuser**(1), **hboot**(1), **killall**(1), **lamhalt**(1), **lamwipe**(1), **ldattach**(8), **lp5250d**(1), **lsof**(8), **miau**(1), **nxtvepg**(1), **pgrep**(1), **pmsignal**(1), **sge_checkpoint**(5), **sge_conf**(5), **sge_queue_conf**(5), **sigaction**(2), **signal**(2), **skill**(1), **tcpdump**(8), **tgif**(1), **thttpd**(8), **timeout**(1), **uucico**(8)

sort(1) - Linux man page

Name

sort - sort lines of text files

Synopsis

sort [*OPTION*]... [*FILE*]...
sort [*OPTION*]... *--files0-from=F*

Description

Write sorted concatenation of all **FILE**(s) to standard output.

Mandatory arguments to long options are mandatory for short options too. Ordering options:

-b, **--ignore-leading-blanks**
ignore leading blanks
-d, **--dictionary-order**
consider only blanks and alphanumeric characters
-f, **--ignore-case**
fold lower case to upper case characters
-g, **--general-numeric-sort**
compare according to general numerical value
-i, **--ignore-nonprinting**
consider only printable characters
-M, **--month-sort**
compare (unknown) < 'JAN' < ... < 'DEC'
-h, **--human-numeric-sort**
compare human readable numbers (e.g., 2K 1G)
-n, **--numeric-sort**
compare according to string numerical value
-R, **--random-sort**
sort by random hash of keys
--random-source=*FILE*
get random bytes from FILE
-r, **--reverse**
reverse the result of comparisons
--sort=*WORD*
sort according to WORD: general-numeric **-g**, human-numeric **-h**, month **-M**, numeric **-n**, random **-R**, version **-V**
-V, **--version-sort**
natural sort of (version) numbers within text

Other options:

--batch-size=*NMERGE*
merge at most NMERGE inputs at once; for more use temp files
-c, **--check**, **--check**=*diagnose-first*
check for sorted input; do not sort
-C, **--check**=*quiet*, **--check**=*silent*
like **-c**, but do not report first bad line
--compress-program=*PROG*
compress temporaries with PROG; decompress them with PROG **-d**
--files0-from=*F*
read input from the files specified by NUL-terminated names in file F; If F is - then read names from standard input
-k, **--key**=*POS1[*,POS2]
start a key at POS1 (origin 1), end it at POS2 (default end of line)
-m, **--merge**
merge already sorted files; do not sort
-o, **--output**=*FILE*
write result to FILE instead of standard output
-s, **--stable**
stabilize sort by disabling last-resort comparison
-S, **--buffer-size**=*SIZE*
use SIZE for main memory buffer
-t, **--field-separator**=*SEP*
use SEP instead of non-blank to blank transition
-T, **--temporary-directory**=*DIR*
use DIR for temporaries, not $TMPDIR or /tmp; multiple options specify multiple directories
-u, **--unique**
with **-c**, check for strict ordering; without **-c**, output only the first of an equal run
-z, **--zero-terminated**
end lines with 0 byte, not newline
--help
display this help and exit
--version
output version information and exit

POS is F[.C][OPTS], where F is the field number and C the character position in the field; both are origin 1. If neither **-t** nor **-b** is in effect, characters in a field are counted from the beginning of the preceding whitespace. OPTS is one or more single-letter ordering options, which override global ordering options for that key. If no key is given, use the entire line as the key.

SIZE may be followed by the following multiplicative suffixes: % 1% of memory, b 1, K 1024 (default), and so on for M, G, T, P, E, Z, Y.

find(1) - Linux man page

Name

find - search for files in a directory hierarchy

Synopsis

find [-H] [-L] [-P] [-D debugopts] [-Olevel] [path...] [expression]

Description

This manual page documents the GNU version of **find**. GNU **find** searches the directory tree rooted at each given file name by evaluating the given expression from left to right, according to the rules of precedence (see section OPERATORS), until the outcome is known (the left hand side is false for *and* operations, true for *or*), at which point **find** moves on to the next file name.

If you are using **find** in an environment where security is important (for example if you are using it to search directories that are writable by other users), you should read the "Security Considerations" chapter of the findutils documentation, which is called **Finding Files** and comes with findutils. That document also includes a lot more detail and discussion than this manual page, so you may find it a more useful source of information.

Options

The **-H**, **-L** and **-P** options control the treatment of symbolic links. Command-line arguments following these are taken to be names of files or directories to be examined, up to the first argument that begins with '-', or the argument '(' or '!'. That argument and any following arguments are taken to be the expression describing what is to be searched for. If no paths are given, the current directory is used. If no expression is given, the expression **-print** is used (but you should probably consider using **-print0** instead, anyway).

This manual page talks about 'options' within the expression list. These options control the behaviour of **find** but are specified immediately after the last path name. The five 'real' options **-H**, **-L**, **-P**, **-D** and **-O** must appear before the first path name, if at all. A double dash **--** can also be used to signal that any remaining arguments are not options (though ensuring that all start points begin with either './' or '/' is generally safer if you use wildcards in the list of start points).

-P

Never follow symbolic links. This is the default behaviour. When **find** examines or prints information a file, and the file is a symbolic link, the information used shall be taken from the properties of the symbolic link itself.

-L

Follow symbolic links. When **find** examines or prints information about files, the information used shall be taken from the properties of the file to which the link points, not from the link itself (unless it is a broken symbolic link or **find** is unable to examine the file to which the link points). Use of this option implies **-noleaf**. If you later use the **-P** option, **-noleaf** will still be in effect. If **-L** is in effect and **find** discovers a symbolic link to a subdirectory during its search, the subdirectory pointed to by the symbolic link will be searched.

> When the **-L** option is in effect, the **-type** predicate will always match against the type of the file that a symbolic link points to rather than the link itself (unless the symbolic link is broken). Using **-L** causes the **-lname** and **-ilname** predicates always to return false.

-H

Do not follow symbolic links, except while processing the command line arguments. When **find** examines or prints information about files, the information used shall be taken from the properties of the symbolic link itself. The only exception to this behaviour is when a file specified on the command line is a symbolic link, and the link can be resolved. For that situation, the information used is taken from whatever the link points to (that is, the link is followed). The information about the link itself is used as a fallback if the file pointed to by the symbolic link cannot be examined. If **-H** is in effect and one of the paths specified on the command line is a symbolic link to a directory, the contents of that directory will be examined (though of course -maxdepth 0 would prevent this).

If more than one of **-H**, **-L** and **-P** is specified, each overrides the others; the last one appearing on the command line takes effect. Since it is the default, the **-P** option should be considered to be in effect unless either **-H** or **-L** is specified.

GNU **find** frequently stats files during the processing of the command line itself, before any searching has begun. These options also affect how those arguments are processed. Specifically, there are a number of tests that compare files listed on the command line against a file we are currently considering. In each case, the file specified on the command line will have been examined and some of its properties will have been saved. If the named file is in fact a symbolic link, and the **-P** option is in effect (or if neither **-H** nor **-L** were specified), the information used for the comparison will be taken from the properties of the symbolic link. Otherwise, it will be taken from the properties of the file the link points to. If **find** cannot follow the link (for example because it has insufficient privileges or the link points to a nonexistent file) the properties of the link itself will be used.

When the **-H** or **-L options are in effect, any symbolic links listed** as the argument of **-newer** will be dereferenced, and the timestamp will be taken from the file to which the symbolic link points. The same consideration applies to **-newerXY**, **-anewer** and **-cnewer**.

The **-follow** option has a similar effect to **-L**, though it takes effect at the point where it appears (that is, if **-L** is not used but **-follow** is, any symbolic links appearing after **-follow** on the command line will be dereferenced, and those before it will not).

-D debugoptions

Print diagnostic information; this can be helpful to diagnose problems with why **find** is not doing what you want. The list of debug options should be comma separated. Compatibility of the debug options is not guaranteed between releases of findutils. For a complete list of valid debug options, see the output of **find -D help**. Valid debug options include

help

Explain the debugging options

tree

Show the expression tree in its original and optimised form.

stat

Print messages as files are examined with the **stat** and **lstat** system calls. The **find** program tries to minimise such calls.

opt

Prints diagnostic information relating to the optimisation of the expression tree; see the -O option.

rates

Prints a summary indicating how often each predicate succeeded or failed.

-Olevel

Enables query optimisation. The **find** program reorders tests to speed up execution while preserving the overall effect; that is, predicates with side effects are not reordered relative to each other. The optimisations performed at each optimisation level are as follows.

0

Equivalent to optimisation level 1.

1

This is the default optimisation level and corresponds to the traditional behaviour. Expressions are reordered so that tests based only on the names of files (for example **-name** and **-regex**) are performed first.

2

Any **-type** or **-xtype** tests are performed after any tests based only on the names of files, but before any tests that require information from the inode. On many modern versions of Unix, file types are returned by **readdir()**

and so these predicates are faster to evaluate than predicates which need to stat the file first.

3

At this optimisation level, the full cost-based query optimiser is enabled. The order of tests is modified so that cheap (i.e. fast) tests are performed first and more expensive ones are performed later, if necessary. Within each cost band, predicates are evaluated earlier or later according to whether they are likely to succeed or not. For **-o**, predicates which are likely to succeed are evaluated earlier, and for **-a**, predicates which are likely to fail are evaluated earlier.

The cost-based optimiser has a fixed idea of how likely any given test is to succeed. In some cases the probability takes account of the specific nature of the test (for example, **-type f** is assumed to be more likely to succeed than **-type c**). The cost-based optimiser is currently being evaluated. If it does not actually improve the performance of **find**, it will be removed again. Conversely, optimisations that prove to be reliable, robust and effective may be enabled at lower optimisation levels over time. However, the default behaviour (i.e. optimisation level 1) will not be changed in the 4.3.x release series. The findutils test suite runs all the tests on **find** at each optimisation level and ensures that the result is the same.

Expressions

The expression is made up of options (which affect overall operation rather than the processing of a specific file, and always return true), tests (which return a true or false value), and actions (which have side effects and return a true or false value), all separated by operators. **-and** is assumed where the operator is omitted.

If the expression contains no actions other than **-prune**, **-print** is performed on all files for which the expression is true.

OPTIONS

All options always return true. Except for **-daystart**, **-follow** and **-regextype**, the options affect all tests, including tests specified before the option. This is because the options are processed when the command line is parsed, while the tests don't do anything until files are examined. The **-daystart**, **-follow** and **-regextype** options are different in this respect, and have an effect only on tests which appear later in the command line. Therefore, for clarity, it is best to place them at the beginning of the expression. A warning is issued if you don't do this.

-d

A synonym for -depth, for compatibility with FreeBSD, NetBSD, MacOS X and OpenBSD.

-daystart

Measure times (for **-amin**, **-atime**, **-cmin**, **-ctime**, **-mmin**, and **-mtime**) from the beginning of today rather than from 24 hours ago. This option only affects tests which appear later on the command line.

-depth

Process each directory's contents before the directory itself. The -delete action also implies **-depth**.

-follow

Deprecated; use the **-L** option instead. Dereference symbolic links. Implies **-noleaf**. The **-follow** option affects only those tests which appear after it on the command line. Unless the **-H** or **-L** option has been specified, the position of the **-follow** option changes the behaviour of the **-newer** predicate; any files listed as the argument of **-newer** will be dereferenced if they are symbolic links. The same consideration applies to **-newerXY**, **-anewer** and **-cnewer**. Similarly, the **-type** predicate will always match against the type of the file that a symbolic link points to rather than the link itself. Using **-follow** causes the **-lname and -ilname** predicates always to return false.

-help, --help

Print a summary of the command-line usage of **find** and exit.

-ignore_readdir_race

Normally, **find** will emit an error message when it fails to stat a file. If you give this option and a file is deleted between the time **find** reads the name of the file from the directory and the time it tries to stat the file, no error message will be issued. This also applies to files or directories whose names are given on the command line. This option takes effect at the time the command line is read, which means that you cannot search one part of the

filesystem with this option on and part of it with this option off (if you need to do that, you will need to issue two **find** commands instead, one with the option and one without it).

-maxdepth *levels*

Descend at most *levels* (a non-negative integer) levels of directories below the command line arguments. **-maxdepth 0** means only apply the tests and actions to the command line arguments.

-mindepth *levels*

Do not apply any tests or actions at levels less than *levels* (a non-negative integer). **-mindepth 1** means process all files except the command line arguments.

-mount

Don't descend directories on other filesystems. An alternate name for **-xdev**, for compatibility with some other versions of **find**.

-noignore_readdir_race

Turns off the effect of **-ignore_readdir_race**.

-noleaf

Do not optimize by assuming that directories contain 2 fewer subdirectories than their hard link count. This option is needed when searching filesystems that do not follow the Unix directory-link convention, such as CD-ROM or MS-DOS filesystems or AFS volume mount points. Each directory on a normal Unix filesystem has at least 2 hard links: its name and its '.' entry. Additionally, its subdirectories (if any) each have a '..' entry linked to that directory. When **find** is examining a directory, after it has statted 2 fewer subdirectories than the directory's link count, it knows that the rest of the entries in the directory are non-directories ('leaf' files in the directory tree). If only the files' names need to be examined, there is no need to stat them; this gives a significant increase in search speed.

-regextype *type*

Changes the regular expression syntax understood by **-regex** and **-iregex** tests which occur later on the command line. Currently-implemented types are emacs (this is the default), posix-awk, posix-basic, posix-egrep and posix-extended.

-version, --version

Print the **find** version number and exit.

-warn, -nowarn

Turn warning messages on or off. These warnings apply only to the command line usage, not to any conditions that **find** might encounter when it searches directories. The default behaviour corresponds to **-warn** if standard input is a tty, and to **-nowarn** otherwise.

-xautofs

Don't descend directories on autofs filesystems.

-xdev

Don't descend directories on other filesystems.

TESTS

Some tests, for example **-newerXY** and **-samefile**, allow comparison between the file currently being examined and some reference file specified on the command line. When these tests are used, the interpretation of the reference file is determined by the options **-H**, **-L** and **-P** and any previous **-follow**, but the reference file is only examined once, at the time the command line is parsed. If the reference file cannot be examined (for example, the **stat**(2) system call fails for it), an error message is issued, and **find** exits with a nonzero status.

Numeric arguments can be specified as

+n

for greater than *n*,

-n

for less than *n*,

n

for exactly *n*.

%Z

(SELinux only) file's security context.

A '%' character followed by any other character is discarded, but the other character is printed (don't rely on this, as further format characters may be introduced). A '%' at the end of the format argument causes undefined behaviour since there is no following character. In some locales, it may hide your door keys, while in others it may remove the final page from the novel you are reading.

The %m and %d directives support the **#** , **0** and **+** flags, but the other directives do not, even if they print numbers. Numeric directives that do not support these flags include **G**, **U**, **b**, **D**, **k** and **n**. The '-' format flag is supported and changes the alignment of a field from right-justified (which is the default) to left-justified.

See the **UNUSUAL FILENAMES** section for information about how unusual characters in filenames are handled.

-prune

True; if the file is a directory, do not descend into it. If **-depth** is given, false; no effect. Because **-delete** implies **-depth**, you cannot usefully use **-prune** and **-delete together.**

-quit

Exit immediately. No child processes will be left running, but no more paths specified on the command line will be processed. For example, **find /tmp/foo /tmp/bar -print -quit** will print only **/tmp/foo**. Any command lines which have been built up with **-execdir ... {} +** will be invoked before **find** exits. The exit status may or may not be zero, depending on whether an error has already occurred.

UNUSUAL FILENAMES

Many of the actions of **find** result in the printing of data which is under the control of other users. This includes file names, sizes, modification times and so forth. File names are a potential problem since they can contain any character except '\0' and '/'. Unusual characters in file names can do unexpected and often undesirable things to your terminal (for example, changing the settings of your function keys on some terminals). Unusual characters are handled differently by various actions, as described below.

-print0, -fprint0

Always print the exact filename, unchanged, even if the output is going to a terminal.

-ls, -fls

Unusual characters are always escaped. White space, backslash, and double quote characters are printed using C-style escaping (for example '\f', '\"'). Other unusual characters are printed using an octal escape. Other printable characters (for **-ls** and **-fls** these are the characters between octal 041 and 0176) are printed as-is.

-printf, -fprintf

If the output is not going to a terminal, it is printed as-is. Otherwise, the result depends on which directive is in use. The directives %D, %F, %g, %G, %H, %Y, and %y expand to values which are not under control of files' owners, and so are printed as-is. The directives %a, %b, %c, %d, %i, %k, %m, %M, %n, %s, %t, %u and %U have values which are under the control of files' owners but which cannot be used to send arbitrary data to the terminal, and so these are printed as-is. The directives %f, %h, %l, %p and %P are quoted. This quoting is performed in the same way as for GNU **ls**. This is not the same quoting mechanism as the one used for **-ls** and **-fls**. If you are able to decide what format to use for the output of **find** then it is normally better to use '\0' as a terminator than to use newline, as file names can contain white space and newline characters. The setting of the 'LC_CTYPE' environment variable is used to determine which characters need to be quoted.

-print, -fprint

Quoting is handled in the same way as for **-printf** and **-fprintf**. If you are using **find** in a script or in a situation where the matched files might have arbitrary names, you should consider using **-print0** instead of **-print**.

The **-ok** and **-okdir** actions print the current filename as-is. This may change in a future release.

OPERATORS

Listed in order of decreasing precedence:

(*expr*)

Force precedence. Since parentheses are special to the shell, you will normally need to quote them. Many of the examples in this manual page use backslashes for this purpose: '\(...\)' instead of '(...)'.

! *expr*

True if *expr* is false. This character will also usually need protection from interpretation by the shell.

-not *expr*
: Same as ! *expr*, but not POSIX compliant.

expr1 expr2
: Two expressions in a row are taken to be joined with an implied "and"; *expr2* is not evaluated if *expr1* is false.

expr1 -a *expr2*
: Same as *expr1 expr2*.

expr1 -and *expr2*
: Same as *expr1 expr2*, but not POSIX compliant.

expr1 -o *expr2*
: Or; *expr2* is not evaluated if *expr1* is true.

expr1 -or *expr2*
: Same as *expr1* **-o** *expr2*, but not POSIX compliant.

expr1 , *expr2*
: List; both *expr1* and *expr2* are always evaluated. The value of *expr1* is discarded; the value of the list is the value of *expr2*. The comma operator can be useful for searching for several different types of thing, but traversing the filesystem hierarchy only once. The **-fprintf** action can be used to list the various matched items into several different output files.

Standards Conformance

For closest compliance to the POSIX standard, you should set the POSIXLY_CORRECT environment variable. The following options are specified in the POSIX standard (IEEE Std 1003.1, 2003 Edition):

-H

This option is supported.

-L

This option is supported.

-name

This option is supported, but POSIX conformance depends on the POSIX conformance of the system's **fnmatch**(3) library function. As of findutils-4.2.2, shell metacharacters ('*', '?' or '[]' for example) will match a leading '.', because IEEE PASC interpretation 126 requires this. This is a change from previous versions of findutils.

-type

Supported. POSIX specifies 'b', 'c', 'd', 'l', 'p', 'f' and 's'. GNU find also supports 'D', representing a Door, where the OS provides these.

-ok

Supported. Interpretation of the response is according to the "yes" and "no" patterns selected by setting the 'LC_MESSAGES' environment variable. When the 'POSIXLY_CORRECT' environment variable is set, these patterns are taken system's definition of a positive (yes) or negative (no) response. See the system's documentation for **nl_langinfo**(3), in particular YESEXPR and NOEXPR. When 'POSIXLY_CORRECT' is not set, the patterns are instead taken from **find**'s own message catalogue.

-newer

Supported. If the file specified is a symbolic link, it is always dereferenced. This is a change from previous behaviour, which used to take the relevant time from the symbolic link; see the HISTORY section below.

-perm

Supported. If the POSIXLY_CORRECT environment variable is not set, some mode arguments (for example +a+x) which are not valid in POSIX are supported for backward-compatibility.

Other predicates

The predicates **-atime**, **-ctime**, **-depth**, **-group**, **-links**, **-mtime**, **-nogroup**, **-nouser**, **-print**, **-prune**, **-size**, **-user** and **-xdev** are all supported.

The POSIX standard specifies parentheses '(', ')', negation '!' and the 'and' and 'or' operators (**-a**, **-o**).

All other options, predicates, expressions and so forth are extensions beyond the POSIX standard. Many of these extensions are not unique to GNU find, however.

The POSIX standard requires that **find** detects loops:

> The **find** utility shall detect infinite loops; that is, entering a previously visited directory that is an ancestor of the last file encountered. When it detects an infinite loop, find shall write a diagnostic message to standard error and shall either recover its position in the hierarchy or terminate.

GNU **find** complies with these requirements. The link count of directories which contain entries which are hard links to an ancestor will often be lower than they otherwise should be. This can mean that GNU find will sometimes optimise away the visiting of a subdirectory which is actually a link to an ancestor. Since **find** does not actually enter such a subdirectory, it is allowed to avoid emitting a diagnostic message. Although this behaviour may be somewhat confusing, it is unlikely that anybody actually depends on this behaviour. If the leaf optimisation has been turned off with **-noleaf**, the directory entry will always be examined and the diagnostic message will be issued where it is appropriate. Symbolic links cannot be used to create filesystem cycles as such, but if the **-L** option or the **-follow** option is in use, a diagnostic message is issued when **find** encounters a loop of symbolic links. As with loops containing hard links, the leaf optimisation will often mean that **find** knows that it doesn't need to call *stat()* or *chdir()* on the symbolic link, so this diagnostic is frequently not necessary.

The **-d** option is supported for compatibility with various BSD systems, but you should use the POSIX-compliant option **-depth** instead.

The POSIXLY_CORRECT environment variable does not affect the behaviour of the **-regex** or **-iregex** tests because those tests aren't specified in the POSIX standard.

Environment Variables

LANG

Provides a default value for the internationalization variables that are unset or null.

LC_ALL

If set to a non-empty string value, override the values of all the other internationalization variables.

LC_COLLATE

The POSIX standard specifies that this variable affects the pattern matching to be used for the **-name** option. GNU find uses the **fnmatch**(3) library function, and so support for 'LC_COLLATE' depends on the system library. This variable also affects the interpretation of the response to **-ok;** while the 'LC_MESSAGES' variable selects the actual pattern used to interpret the response to **-ok**, the interpretation of any bracket expressions in the pattern will be affected by 'LC_COLLATE'.

LC_CTYPE

This variable affects the treatment of character classes used in regular expressions and also with the **-name** test, if the system's **fnmatch**(3) library function supports this. This variable also affects the interpretation of any character classes in the regular expressions used to interpret the response to the prompt issued by **-ok**. The 'LC_CTYPE' environment variable will also affect which characters are considered to be unprintable when filenames are printed; see the section UNUSUAL FILENAMES.

LC_MESSAGES

Determines the locale to be used for internationalised messages. If the 'POSIXLY_CORRECT' environment variable is set, this also determines the interpretation of the response to the prompt made by the **-ok** action.

NLSPATH

Determines the location of the internationalisation message catalogues.

PATH

Affects the directories which are searched to find the executables invoked by **-exec**, **-execdir**, **-ok** and **-okdir**.

POSIXLY_CORRECT

Determines the block size used by **-ls** and **-fls**. If **POSIXLY_CORRECT** is set, blocks are units of 512 bytes. Otherwise they are units of 1024 bytes.
Setting this variable also turns off warning messages (that is, implies **-nowarn**) by default, because POSIX requires that apart from the output for **-ok**, all messages printed on stderr are diagnostics and must result in a non-zero exit status.

When POSIXLY_CORRECT is not set, **-perm** +zzz is treated just like **-perm** /zzz if +zzz is not a valid symbolic mode. When POSIXLY_CORRECT is set, such constructs are treated as an error.

When POSIXLY_CORRECT is set, the response to the prompt made by the **-ok** action is interpreted according to the system's message catalogue, as opposed to according to **find**'s own message translations.

TZ

Affects the time zone used for some of the time-related format directives of **-printf** and **-fprintf**.

Examples

```
find /tmp -name core -type f -print | xargs /bin/rm -f
```

Find files named **core** in or below the directory **/tmp** and delete them. Note that this will work incorrectly if there are any filenames containing newlines, single or double quotes, or spaces.

find /tmp -name core -type f -print0 | xargs -0 /bin/rm -f

Find files named **core** in or below the directory **/tmp** and delete them, processing filenames in such a way that file or directory names containing single or double quotes, spaces or newlines are correctly handled. The **-name** test comes before the **-type** test in order to avoid having to call **stat(2)** on every file.

```
find . -type f -exec file '{}' \;
```

Runs 'file' on every file in or below the current directory. Notice that the braces are enclosed in single quote marks to protect them from interpretation as shell script punctuation. The semicolon is similarly protected by the use of a backslash, though single quotes could have been used in that case also.

```
find / \
\( -perm -4000 -fprintf /root/suid.txt %#m %u %p\n \) , \
\( -size +100M -fprintf /root/big.txt %-10s %p\n \)
```

Traverse the filesystem just once, listing setuid files and directories into **/root/suid.txt** and large files into **/root/big.txt**.

```
find $HOME -mtime 0
```

Search for files in your home directory which have been modified in the last twenty-four hours. This command works this way because the time since each file was last modified is divided by 24 hours and any remainder is discarded. That means that to match **-mtime 0**, a file will have to have a modification in the past which is less than 24 hours ago.

```
find /sbin /usr/sbin -executable \! -readable -print
```

Search for files which are executable but not readable.

```
find . -perm 664
```

Search for files which have read and write permission for their owner, and group, but which other users can read but not write to. Files which meet these criteria but have other permissions bits set (for example if someone can execute

the file) will not be matched.

```
find . -perm -664
```

Search for files which have read and write permission for their owner and group, and which other users can read, without regard to the presence of any extra permission bits (for example the executable bit). This will match a file which has mode 0777, for example.

```
find . -perm /222
```

Search for files which are writable by somebody (their owner, or their group, or anybody else).

```
find . -perm /220
find . -perm /u+w,g+w
find . -perm /u=w,g=w
```

All three of these commands do the same thing, but the first one uses the octal representation of the file mode, and the other two use the symbolic form. These commands all search for files which are writable by either their owner or their group. The files don't have to be writable by both the owner and group to be matched; either will do.

```
find . -perm -220
find . -perm -g+w,u+w
```

Both these commands do the same thing; search for files which are writable by both their owner and their group.

```
find . -perm -444 -perm /222 ! -perm /111
find . -perm -a+r -perm /a+w ! -perm /a+x
```

These two commands both search for files that are readable for everybody (**-perm -444** or **-perm -a+r**), have at least one write bit set (**-perm /222** or **-perm /a+w**) but are not executable for anybody (**! -perm /111** and **! -perm /a+x** respectively).

```
cd /source-dir
find . -name .snapshot -prune -o \( \! -name *~ -print0 \)|
cpio -pmd0 /dest-dir
```

This command copies the contents of **/source-dir** to **/dest-dir**, but omits files and directories named **.snapshot** (and anything in them). It also omits files or directories whose name ends in **~**, but not their contents. The construct **-prune -o \(... -print0 \)** is quite common. The idea here is that the expression before **-prune** matches things which are to be pruned. However, the **-prune** action itself returns true, so the following **-o** ensures that the right hand side is evaluated only for those directories which didn't get pruned (the contents of the pruned directories are not even visited, so their contents are irrelevant). The expression on the right hand side of the **-o** is in parentheses only for clarity. It emphasises that the **-print0** action takes place only for things that didn't have **-prune** applied to them. Because the default 'and' condition between tests binds more tightly than **-o**, this is the default anyway, but the parentheses help to show what is going on.

```
find repo/ -exec test -d {}/.svn -o -d {}/.git -o -d {}/CVS ; \
-print -prune
```

Given the following directory of projects and their associated SCM administrative directories, perform an efficient search for the projects' roots:

```
repo/project1/CVS
repo/gnu/project2/.svn
repo/gnu/project3/.svn
repo/gnu/project3/src/.svn
```

sed(1) - Linux man page

Name

sed - stream editor for filtering and transforming text

Synopsis

sed [*OPTION*]... *{script-only-if-no-other-script}* [*input-file*]...

Description

Sed is a stream editor. A stream editor is used to perform basic text transformations on an input stream (a file or input from a pipeline). While in some ways similar to an editor which permits scripted edits (such as *ed*), *sed* works by making only one pass over the **input**(s), and is consequently more efficient. But it is *sed*'s ability to filter text in a pipeline which particularly distinguishes it from other types of editors.

-n, **--quiet**, **--silent**
: suppress automatic printing of pattern space

-e script, **--expression**=*script*
: add the script to the commands to be executed

-f script-file, **--file**=*script-file*
: add the contents of script-file to the commands to be executed

--follow-symlinks
: follow symlinks when processing in place; hard links will still be broken.

-i[SUFFIX], **--in-place**[=*SUFFIX*]
: edit files in place (makes backup if extension supplied). The default operation mode is to break symbolic and hard links. This can be changed with **--follow-symlinks** and **--copy**.

-c, **--copy**
: use copy instead of rename when shuffling files in **-i** mode. While this will avoid breaking links (symbolic or hard), the resulting editing operation is not atomic. This is rarely the desired mode; **--follow-symlinks** is usually enough, and it is both faster and more secure.

-l N, **--line-length**=*N*
: specify the desired line-wrap length for the 'l' command

--posix
: disable all GNU extensions.

-r, **--regexp-extended**
: use extended regular expressions in the script.

-s, **--separate**
: consider files as separate rather than as a single continuous long stream.

-u, **--unbuffered**

comm(1) - Linux man page

Name

comm - compare two sorted files line by line

Synopsis

comm [*OPTION*]... *FILE1 FILE2*

Description

Compare sorted files FILE1 and FILE2 line by line.

With no options, produce three-column output. Column one contains lines unique to FILE1, column two contains lines unique to FILE2, and column three contains lines common to both files.

-1
: suppress column 1 (lines unique to FILE1)

-2
: suppress column 2 (lines unique to FILE2)

-3
: suppress column 3 (lines that appear in both files)

--check-order
: check that the input is correctly sorted, even if all input lines are pairable

--nocheck-order
: do not check that the input is correctly sorted

--output-delimiter=*STR*
: separate columns with STR

--help
: display this help and exit

--version
: output version information and exit

Note, comparisons honor the rules specified by 'LC_COLLATE'.

Examples

comm **-12** file1 file2
: Print only lines present in both file1 and file2.

comm **-3**
: file1 file2 Print lines in file1 not in file2, and vice versa.

Author

tr(1) - Linux man page

Name

tr - translate or delete characters

Synopsis

tr [*OPTION*]... *SET1* [*SET2*]

Description

Translate, squeeze, and/or delete characters from standard input, writing to standard output.

-c, **-C**, **--complement**
use the complement of SET1
-d, **--delete**
delete characters in SET1, do not translate
-s, **--squeeze-repeats**
replace each input sequence of a repeated character that is listed in SET1 with a single occurrence of that character
-t, **--truncate-set1**
first truncate SET1 to length of SET2
--help
display this help and exit
--version
output version information and exit

SETs are specified as strings of characters. Most represent themselves. Interpreted sequences are:

\NNN
character with octal value NNN (1 to 3 octal digits)
\\
backslash
\a
audible BEL
\b
backspace
\f
form feed
\n
new line
\r

return

\t
horizontal tab

\v
vertical tab

CHAR1-CHAR2
all characters from CHAR1 to CHAR2 in ascending order

[CHAR*]
in SET2, copies of CHAR until length of SET1

[CHAR*REPEAT]
REPEAT copies of CHAR, REPEAT octal if starting with 0

[:alnum:]
all letters and digits

[:alpha:]
all letters

[:blank:]
all horizontal whitespace

[:cntrl:]
all control characters

[:digit:]
all digits

[:graph:]
all printable characters, not including space

[:lower:]
all lower case letters

[:print:]
all printable characters, including space

[:punct:]
all punctuation characters

[:space:]
all horizontal or vertical whitespace

[:upper:]
all upper case letters

[:xdigit:]
all hexadecimal digits

[=CHAR=]
all characters which are equivalent to CHAR

Translation occurs if **-d** is not given and both SET1 and SET2 appear. **-t** may be used only when translating. SET2 is extended to length of SET1 by repeating its last character as necessary. Excess characters of SET2 are ignored. Only [:lower:] and [:upper:] are guaranteed to expand in ascending order; used in SET2 while translating, they may only be used in pairs to specify case conversion. **-s** uses SET1 if not translating nor deleting; else squeezing uses SET2 and occurs after translation or deletion.

diff(1) - Linux man page

Name

diff - compare files line by line

Synopsis

diff [*OPTION*]... *FILES*

Description

Compare files line by line.

-i --ignore-case
Ignore case differences in file contents.
--ignore-file-name-case
Ignore case when comparing file names.
--no-ignore-file-name-case
Consider case when comparing file names.
-E --ignore-tab-expansion
Ignore changes due to tab expansion.
-b --ignore-space-change
Ignore changes in the amount of white space.
-w --ignore-all-space
Ignore all white space.
-B --ignore-blank-lines
Ignore changes whose lines are all blank.
-I RE **--ignore-matching-lines**=*RE*
Ignore changes whose lines all match RE.
--strip-trailing-cr
Strip trailing carriage return on input.
-a --text
Treat all files as text.
-c -C NUM **--context**[=*NUM*]
Output NUM (default 3) lines of copied context.
-u -U NUM **--unified**[=*NUM*]
Output NUM (default 3) lines of unified context.
--label LABEL
Use LABEL instead of file name.
-p --show-c-function
Show which C function each change is in.
-F RE **--show-function-line**=*RE*
Show the most recent line matching RE.
-q --brief

Output only whether files differ.

-e --ed

Output an ed script.

--normal

Output a normal diff.

-n --rcs

Output an RCS format diff.

-y --side-by-side

Output in two columns.

-W NUM **--width**=*NUM*

Output at most NUM (default 130) print columns.

--left-column

Output only the left column of common lines.

--suppress-common-lines

Do not output common lines.

-D NAME **--ifdef**=*NAME*

Output merged file to show '#ifdef NAME' diffs.

--GTYPE-group-format=*GFMT*

Similar, but format GTYPE input groups with GFMT.

--line-format=*LFMT*

Similar, but format all input lines with LFMT.

--LTYPE-line-format=*LFMT*

Similar, but format LTYPE input lines with LFMT.

LTYPE is 'old', 'new', or 'unchanged'.

GTYPE is LTYPE or 'changed'.

GFMT may contain:

%<

lines from FILE1

%>

lines from FILE2

%=

lines common to FILE1 and FILE2

%[-][WIDTH][.[PREC]]{doxX}LETTER

printf-style spec for LETTER

LETTERs are as follows for new group, lower case for old group:

F

first line number

L

last line number

N

number of lines = L-F+1

E

F-1

M

L+1

LFMT may contain:

%L

contents of line

%l

contents of line, excluding any trailing newline

%[-][WIDTH][.[PREC]]{doxX}n

printf-style spec for input line number

Either GFMT or LFMT may contain:

%%

%

%c'C'

the single character C

%c'\OOO'

the character with octal code OOO

-l --paginate

Pass the output through 'pr' to paginate it.

-t --expand-tabs

Expand tabs to spaces in output.

-T --initial-tab

Make tabs line up by prepending a tab.

-r --recursive

Recursively compare any subdirectories found.

-N --new-file

Treat absent files as empty.

--unidirectional-new-file

Treat absent first files as empty.

-s --report-identical-files

Report when two files are the same.

-x PAT **--exclude**=*PAT*

Exclude files that match PAT.

-X FILE **--exclude-from**=*FILE*

Exclude files that match any pattern in FILE.

-S FILE **--starting-file**=*FILE*

Start with FILE when comparing directories.

--from-file=*FILE1*

Compare FILE1 to all operands. FILE1 can be a directory.

--to-file=*FILE2*

Compare all operands to FILE2. FILE2 can be a directory.

--horizon-lines=*NUM*

Keep NUM lines of the common prefix and suffix.

-d --minimal

Try hard to find a smaller set of changes.

--speed-large-files

grep(1) - Linux man page

Name

grep, egrep, fgrep - print lines matching a pattern

Synopsis

grep [*OPTIONS*] *PATTERN* [*FILE*...]
grep [*OPTIONS*] [**-e** *PATTERN* | **-f** *FILE*] [*FILE*...]

Description

grep searches the named input *FILE*s (or standard input if no files are named, or if a single hyphen-minus (**-**) is given as file name) for lines containing a match to the given *PATTERN*. By default, **grep** prints the matching lines.

In addition, two variant programs **egrep** and **fgrep** are available. **egrep** is the same as **grep -E**. **fgrep** is the same as **grep -F**. Direct invocation as either **egrep** or **fgrep** is deprecated, but is provided to allow historical applications that rely on them to run unmodified.

Options

Generic Program Information

--help

Print a usage message briefly summarizing these command-line options and the bug-reporting address, then exit.

-V, **--version**
Print the version number of **grep** to the standard output stream. This version number should be included in all bug reports (see below).

Matcher Selection

-E, **--extended-regexp**
Interpret *PATTERN* as an extended regular expression (ERE, see below). (**-E** is specified by POSIX .)
-F, **--fixed-strings**
Interpret *PATTERN* as a list of fixed strings, separated by newlines, any of which is to be matched. (**-F** is specified by POSIX .)
-G, **--basic-regexp**
Interpret *PATTERN* as a basic regular expression (BRE, see below). This is the default.
-P, **--perl-regexp**

Interpret *PATTERN* as a Perl regular expression. This is highly experimental and **grep -P** may warn of unimplemented features.

Matching Control

-e *PATTERN*, **--regexp=***PATTERN*
Use *PATTERN* as the pattern. This can be used to specify multiple search patterns, or to protect a pattern beginning with a hyphen (**-**). (**-e** is specified by POSIX .)

-f *FILE*, **--file=***FILE*
Obtain patterns from *FILE*, one per line. The empty file contains zero patterns, and therefore matches nothing. (**-f** is specified by POSIX .)

-i, **--ignore-case**
Ignore case distinctions in both the *PATTERN* and the input files. (**-i** is specified by POSIX .)

-v, **--invert-match**
Invert the sense of matching, to select non-matching lines. (**-v** is specified by POSIX .)

-w, **--word-regexp**
Select only those lines containing matches that form whole words. The test is that the matching substring must either be at the beginning of the line, or preceded by a non-word constituent character. Similarly, it must be either at the end of the line or followed by a non-word constituent character. Word-constituent characters are letters, digits, and the underscore.

-x, **--line-regexp**
Select only those matches that exactly match the whole line. (**-x** is specified by POSIX .)

-y

Obsolete synonym for **-i**.

General Output Control

-c, **--count**
Suppress normal output; instead print a count of matching lines for each input file. With the **-v**, **--invert-match** option (see below), count non-matching lines. (**-c** is specified by POSIX .)

--color[=*WHEN*], **--colour**[=*WHEN*]
Surround the matched (non-empty) strings, matching lines, context lines, file names, line numbers, byte offsets, and separators (for fields and groups of context lines) with escape sequences to display them in color on the terminal. The colors are defined by the environment variable **GREP_COLORS**. The deprecated environment variable **GREP_COLOR** is still supported, but its setting does not have priority. *WHEN* is **never**, **always**, or **auto**.

-L, **--files-without-match**
Suppress normal output; instead print the name of each input file from which no output would normally have been printed. The scanning will stop on the first match.

-l, **--files-with-matches**
Suppress normal output; instead print the name of each input file from which output would normally have been printed. The scanning will stop on the first match. (**-l** is specified by POSIX .)

-m *NUM*, **--max-count=***NUM*
Stop reading a file after *NUM* matching lines. If the input is standard input from a regular file, and *NUM* matching lines are output, **grep** ensures that the standard input is positioned to just after the last matching line before exiting, regardless of the presence of trailing context lines. This enables a calling process to resume a search. When **grep** stops after *NUM* matching lines, it outputs any trailing context lines. When the **-c** or **--count** option is also used, **grep** does not output a count greater than *NUM*. When the **-v** or **--invert-match** option is also used, **grep** stops after outputting *NUM* non-matching lines.

-o, **--only-matching**
Print only the matched (non-empty) parts of a matching line, with each such part on a separate output line.

-q, **--quiet**, **--silent**
Quiet; do not write anything to standard output. Exit immediately with zero status if any match is found, even if an error was detected. Also see the **-s** or **--no-messages** option. (**-q** is specified by POSIX .)

-s, **--no-messages**
Suppress error messages about nonexistent or unreadable files. Portability note: unlike GNU **grep**, 7th Edition Unix **grep** did not conform to POSIX , because it lacked **-q** and its **-s** option behaved like GNU **grep**'s **-q** option. USG -style **grep** also lacked **-q** but its **-s** option behaved like GNU **grep**. Portable shell scripts should avoid both **-q** and **-s** and should redirect standard and error output to **/dev/null** instead. (**-s** is specified by POSIX .)

Output Line Prefix Control

-b, **--byte-offset**
Print the 0-based byte offset within the input file before each line of output. If **-o** (**--only-matching**) is specified, print the offset of the matching part itself.

-H, **--with-filename**
Print the file name for each match. This is the default when there is more than one file to search.

-h, **--no-filename**
Suppress the prefixing of file names on output. This is the default when there is only one file (or only standard input) to search.

--label=*LABEL*
Display input actually coming from standard input as input coming from file *LABEL*. This is especially useful when implementing tools like **zgrep**, e.g., **gzip -cd foo.gz | grep --label=foo -H something**. See also the **-H** option.

-n, **--line-number**

Prefix each line of output with the 1-based line number within its input file. (**-n** is specified by POSIX .)

-T, **--initial-tab**

Make sure that the first character of actual line content lies on a tab stop, so that the alignment of tabs looks normal. This is useful with options that prefix their output to the actual content: **-H**,**-n**, and **-b**. In order to improve the probability that lines from a single file will all start at the same column, this also causes the line number and byte offset (if present) to be printed in a minimum size field width.

-u, **--unix-byte-offsets**

Report Unix-style byte offsets. This switch causes **grep** to report byte offsets as if the file were a Unix-style text file, i.e., with CR characters stripped off. This will produce results identical to running **grep** on a Unix machine. This option has no effect unless **-b** option is also used; it has no effect on platforms other than MS-DOS and MS-Windows.

-Z, **--null**

Output a zero byte (the ASCII **NUL** character) instead of the character that normally follows a file name. For example, **grep -lZ** outputs a zero byte after each file name instead of the usual newline. This option makes the output unambiguous, even in the presence of file names containing unusual characters like newlines. This option can be used with commands like **find -print0**, **perl -0**, **sort -z**, and **xargs -0** to process arbitrary file names, even those that contain newline characters.

Context Line Control

-A *NUM*, **--after-context=***NUM*

Print *NUM* lines of trailing context after matching lines. Places a line containing a group separator (**--**) between contiguous groups of matches. With the **-o** or **--only-matching** option, this has no effect and a warning is given.

-B *NUM*, **--before-context=***NUM*

Print *NUM* lines of leading context before matching lines. Places a line containing a group separator (**--**) between contiguous groups of matches. With the **-o** or **--only-matching** option, this has no effect and a warning is given.

-C *NUM*, **-***NUM*, **--context=***NUM*

Print *NUM* lines of output context. Places a line containing a group separator (**--**) between contiguous groups of matches. With the **-o** or **--only-matching** option, this has no effect and a warning is given.

File and Directory Selection

-a, **--text**

Process a binary file as if it were text; this is equivalent to the **--binary-files=text** option.

--binary-files=*TYPE*

If the first few bytes of a file indicate that the file contains binary data, assume that the file is of type *TYPE*. By default, *TYPE* is **binary**, and **grep** normally outputs either

a one-line message saying that a binary file matches, or no message if there is no match. If *TYPE* is **without-match**, **grep** assumes that a binary file does not match; this is equivalent to the **-I** option. If *TYPE* is **text**, **grep** processes a binary file as if it were text; this is equivalent to the **-a** option. *Warning:* **grep --binary-files=text** might output binary garbage, which can have nasty side effects if the output is a terminal and if the terminal driver interprets some of it as commands.

-D *ACTION*, **--devices=***ACTION*

If an input file is a device, FIFO or socket, use *ACTION* to process it. By default, *ACTION* is **read**, which means that devices are read just as if they were ordinary files. If *ACTION* is **skip**, devices are silently skipped.

-d *ACTION*, **--directories=***ACTION*

If an input file is a directory, use *ACTION* to process it. By default, *ACTION* is **read**, which means that directories are read just as if they were ordinary files. If *ACTION* is **skip**, directories are silently skipped. If *ACTION* is **recurse**, **grep** reads all files under each directory, recursively; this is equivalent to the **-r** option.

--exclude=*GLOB*

Skip files whose base name matches *GLOB* (using wildcard matching). A file-name glob can use *****, **?**, and **[**...**]** as wildcards, and **** to quote a wildcard or backslash character literally.

--exclude-from=*FILE*

Skip files whose base name matches any of the file-name globs read from *FILE* (using wildcard matching as described under **--exclude**).

--exclude-dir=*DIR*

Exclude directories matching the pattern *DIR* from recursive searches.

-I

Process a binary file as if it did not contain matching data; this is equivalent to the **--binary-files=without-match** option.

--include=*GLOB*

Search only files whose base name matches *GLOB* (using wildcard matching as described under **--exclude**).

-R, **-r**, **--recursive**

Read all files under each directory, recursively; this is equivalent to the **-d recurse** option.

Other Options

--line-buffered

Use line buffering on output. This can cause a performance penalty.

--mmap

If possible, use the **mmap**(2) system call to read input, instead of the default **read**(2) system call. In some situations, **--mmap** yields better performance. However, **--mmap**

can cause undefined behavior (including core dumps) if an input file shrinks while **grep** is operating, or if an I/O error occurs.

-U, **--binary**
Treat the **file**(s) as binary. By default, under MS-DOS and MS -Windows, **grep** guesses the file type by looking at the contents of the first 32KB read from the file. If **grep** decides the file is a text file, it strips the CR characters from the original file contents (to make regular expressions with **^** and **$** work correctly). Specifying **-U** overrules this guesswork, causing all files to be read and passed to the matching mechanism verbatim; if the file is a text file with CR/LF pairs at the end of each line, this will cause some regular expressions to fail. This option has no effect on platforms other than MS-DOS and MS -Windows.

-z, **--null-data**
Treat the input as a set of lines, each terminated by a zero byte (the ASCII **NUL** character) instead of a newline. Like the **-Z** or **--null** option, this option can be used with commands like **sort -z** to process arbitrary file names.

Regular Expressions

A regular expression is a pattern that describes a set of strings. Regular expressions are constructed analogously to arithmetic expressions, by using various operators to combine smaller expressions.

grep understands three different versions of regular expression syntax: "basic," "extended" and "perl." In GNU **grep**, there is no difference in available functionality between basic and extended syntaxes. In other implementations, basic regular expressions are less powerful. The following description applies to extended regular expressions; differences for basic regular expressions are summarized afterwards. Perl regular expressions give additional functionality, and are documented in **pcresyntax**(3) and **pcrepattern**(3), but may not be available on every system.

The fundamental building blocks are the regular expressions that match a single character. Most characters, including all letters and digits, are regular expressions that match themselves. Any meta-character with special meaning may be quoted by preceding it with a backslash.

The period **.** matches any single character.

Character Classes and Bracket Expressions

A *bracket expression* is a list of characters enclosed by **[** and **]**. It matches any single character in that list; if the first character of the list is the caret **^** then it matches any character *not* in the list. For example, the regular expression **[0123456789]** matches any single digit.

Within a bracket expression, a *range expression* consists of two characters separated by a hyphen. It matches any single character that sorts between the two characters, inclusive, using the locale's collating sequence and character set. For example, in the default C locale, **[a-d]** is equivalent to **[abcd]**. Many locales sort characters in dictionary order, and in these locales **[a-d]** is typically not equivalent to **[abcd]**; it might be equivalent to **[aBbCcDd]**, for example. To obtain the traditional interpretation of bracket expressions, you can use the C locale by setting the **LC_ALL** environment variable to the value **C**.

Finally, certain named classes of characters are predefined within bracket expressions, as follows. Their names are self explanatory, and they are **[:alnum:]**, **[:alpha:]**, **[:cntrl:]**, **[:digit:]**, **[:graph:]**, **[:lower:]**, **[:print:]**, **[:punct:]**, **[:space:]**, **[:upper:]**, and **[:xdigit:].** For example, **[[:alnum:]]** means **[0-9A-Za-z]**, except the latter form depends upon the C locale and the ASCII character encoding, whereas the former is independent of locale and character set. (Note that the brackets in these class names are part of the symbolic names, and must be included in addition to the brackets delimiting the bracket expression.) Most meta-characters lose their special meaning inside bracket expressions. To include a literal **]** place it first in the list. Similarly, to include a literal **^** place it anywhere but first. Finally, to include a literal **-** place it last.

Anchoring

The caret **^** and the dollar sign **$** are meta-characters that respectively match the empty string at the beginning and end of a line.

The Backslash Character and Special Expressions

The symbols **\<** and **\>** respectively match the empty string at the beginning and end of a word. The symbol **\b** matches the empty string at the edge of a word, and **\B** matches the empty string provided it's *not* at the edge of a word. The symbol **\w** is a synonym for **[[:alnum:]]** and **\W** is a synonym for **[^[:alnum:]]**.

Repetition

A regular expression may be followed by one of several repetition operators:
?

The preceding item is optional and matched at most once.

*****

The preceding item will be matched zero or more times.

+

The preceding item will be matched one or more times.

{*n*}

The preceding item is matched exactly *n* times.

{*n***,}**

The preceding item is matched *n* or more times.

{,*m***}**

The preceding item is matched at most *m* times.

{*n***,***m***}**

The preceding item is matched at least *n* times, but not more than *m* times.

Concatenation

Two regular expressions may be concatenated; the resulting regular expression matches any string formed by concatenating two substrings that respectively match the concatenated expressions.

Alternation

Two regular expressions may be joined by the infix operator **|**; the resulting regular expression matches any string matching either alternate expression.

Precedence

Repetition takes precedence over concatenation, which in turn takes precedence over alternation. A whole expression may be enclosed in parentheses to override these precedence rules and form a subexpression.

Back References and Subexpressions

The back-reference *****n*, where *n* is a single digit, matches the substring previously matched by the *n*th parenthesized subexpression of the regular expression.

Basic vs Extended Regular Expressions

In basic regular expressions the meta-characters **?**, **+**, **{**, **|**, **(**, and **)** lose their special meaning; instead use the backslashed versions **\?**, **\+**, **\{**, **\|**, **\(**, and **\)**.

Traditional **egrep** did not support the **{** meta-character, and some **egrep** implementations support **\{** instead, so portable scripts should avoid **{** in **grep -E** patterns and should use **[{]** to match a literal **{**.

GNU **grep -E** attempts to support traditional usage by assuming that **{** is not special if it would be the start of an invalid interval specification. For example, the command **grep -E '{1'** searches for the two-character string **{1** instead of reporting a syntax error in the