

Unix Tools for Probing Executable Files

Daniel Topa
HII-TSD
daniel.topa@hii-tds.com

October 6, 2024

Abstract

This article surveys Unix tools for the exploration of executable files, some of which depend upon the application being compiled with debug information. The manual pages are included, making this document usefull in siloed computing networks.

Contents

1	Overview	2
2	Command Examples	2
2.1	ldd	2
2.2	lddconfig	2
2.3	locate	2
2.4	lsof	2
2.4.1	lsof on Process ID	4
2.4.2	lsof on User	4
2.5	objdump	5
2.6	readelf	6
2.7	nm	6
2.8	strace	6
2.8.1	Trace System Calls To A Given Path	6
2.8.2	Inventory time, calls, and errors for every system call	6
2.8.3	Identify Information Associated With File Descriptorsl	7
2.9	strings	8
3	Manual Pages	8
3.1	ldd: Print Shared Object Dependencies	8
3.2	lddconfig: Configure Dynamic Linker Run-time Bindings	10
3.3	locate: List File in Databases	12
3.4	lsof: Show Open Files	17
3.5	objdump: Display Information From Object Files	76
3.6	readelf: Display Information On ELF Files	96
3.7	nm: List Symbols From Object Files	108
3.8	strace: Trace System Calls and Signals	117
3.9	strings: Print Sequences Of Printable Characters	144

1 Overview

Here are several Unix commands for probing executable files. The following section shows sample usage for each command and the final section contains the information from the **manual** page.

1. `gdb`
2. `ldd`
3. `lddconfig`
4. `locate`
5. `objdump`
6. `lsof`
7. `readelf`
8. `nm`
9. `strace`
10. `strings`

The goal is to be able to resolve the workings of an executable file exploiting the ELF structure shown in figures 1. The next figure, 2, shows the relationship between source files, header files, shared objects, and the executable program.

2 Command Examples

2.1 `ldd`

The command `ldd` prints shared object dependencies.

```
root@69cb14a32689:/# ldd /bin/bash
linux-vdso.so.1 (0x00007ffe64317000)
libtinfo.so.6 => /lib/x86_64-linux-gnu/libtinfo.so.6 (0x00007f842112d000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f8420f04000)
/lib64/ld-linux-x86-64.so.2 (0x00007f84212e3000)
```

2.2 `lddconfig`

Stub for `lddconfig` in `/sbin/lddconfig`. Configure dynamic linker run-time bindings.

2.3 `locate`

The `locate` command lists files in a prebuilt database of files generated by the `updatedb` command or by a daemon and compressed using incremental encoding.

```
dantopa@92bc4c447e32:/$ locate libc.so.6
/usr/lib/x86_64-linux-gnu/libc.so.6
/usr/lib32/libc.so.6
```

2.4 `lsof`

This command does an `ls` on open files. The example shows how to query both a user and a process id (pid).

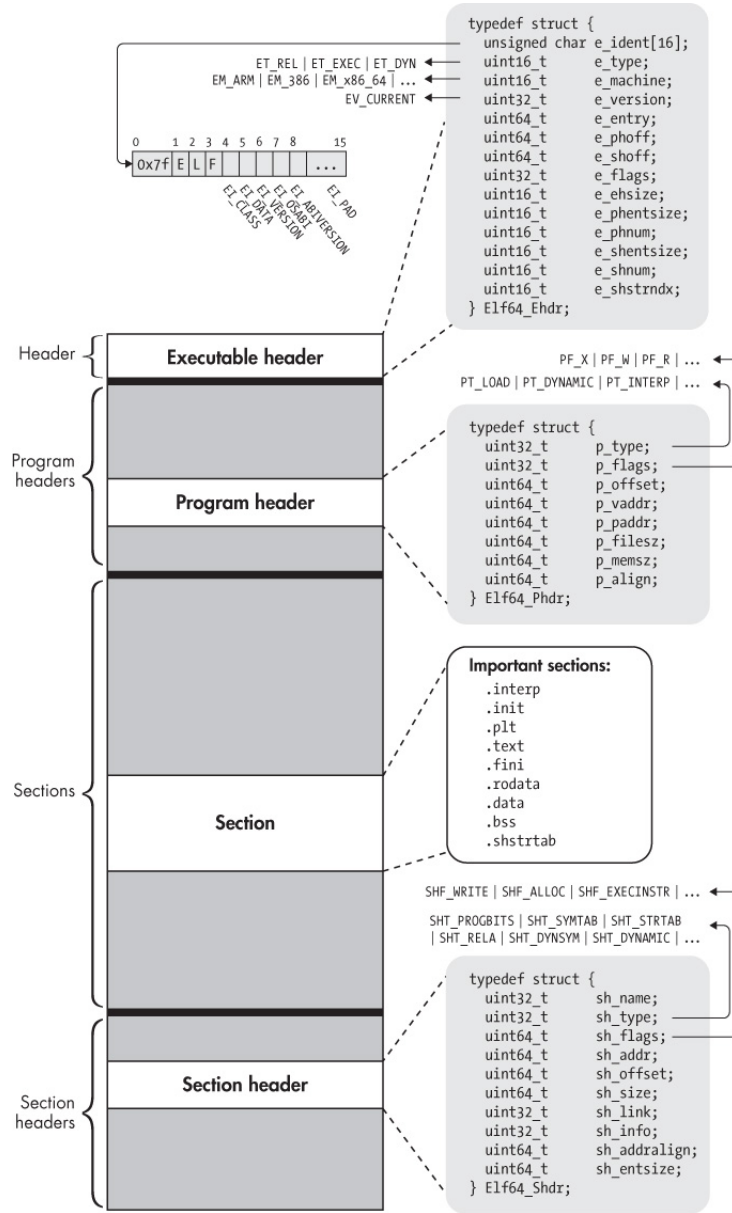


Figure 1: The structure of a Unix ELF file.

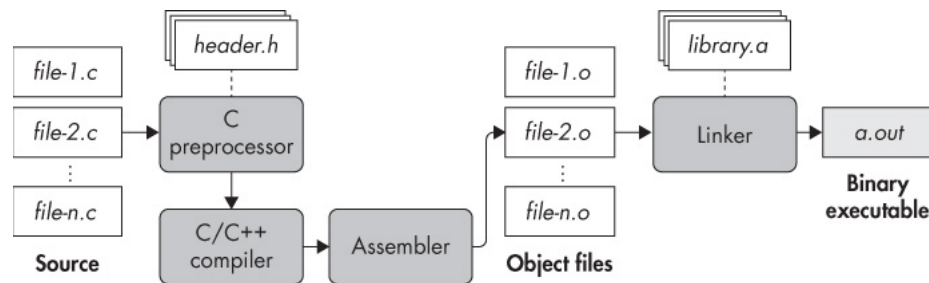


Figure 2: Connecting source files, object files, libraries, and bindary executables.

2.4.1 lsof on Process ID

The `lsdf` command shows open files, here for the `bash` process with `PID = 10932`:

```
dantopa@92bc4c447e32:~$ ps
  PID TTY          TIME CMD
10932 pts/1    00:00:00 bash
11152 pts/1    00:00:00 ps
dantopa@92bc4c447e32:~$ lsof -p 10932
COMMAND    PID    USER    FD  TYPE DEVICE SIZE/OFF  NODE NAME
bash       10932  dantopa  cwd  DIR   0,71    4096  6820049 /home/dantopa
bash       10932  dantopa  rtd  DIR   0,71    4096  61653409 /
bash       10932  dantopa  txt  REG   0,71   1396520 62702252 /usr/bin/bash
bash       10932  dantopa  mem  REG  254,1          62702252 /usr/bin/bash (path dev=0,71)
bash       10932  dantopa  mem  REG  254,1          63095938 /usr/lib/x86_64-linux-gnu/libc.so.6 (path dev=0,71)
bash       10932  dantopa  mem  REG  254,1          1190606 /usr/lib/x86_64-linux-gnu/libtinfo.so.6.3 (path dev=0,71)
bash       10932  dantopa  mem  REG  254,1          63095935 /usr/lib/x86_64-linux-gnu/ld-linux-x86_64.so.2 (path dev=0,71)
bash       10932  dantopa   0u   CHR 136,1          0t0    4 /dev/pts/1
bash       10932  dantopa   1u   CHR 136,1          0t0    4 /dev/pts/1
bash       10932  dantopa   2u   CHR 136,1          0t0    4 /dev/pts/1
bash       10932  dantopa  255u  CHR 136,1          0t0    4 /dev/pts/1
```

2.4.2 lsof on User

These are open files for user dantopa:

```
dantopa@92bc4c447e32:~$ lsuf -u dantopa
COMMAND      PID      USER      FD      TYPE  DEVICE  SIZE/OFF      NODE  NAME
bash         10921   dantopa   cwd      DIR    0,71    4096  61653409  /
bash         10921   dantopa   rtd      DIR    0,71    4096  61653409  /
bash         10921   dantopa   txt      REG    0,71   1396520  62702252  /usr/bin/bash
bash         10921   dantopa   mem      REG   254,1    62702252  /usr/bin/bash (path dev=0,71)
bash         10921   dantopa   mem      REG   254,1    63095938  /usr/lib/x86_64-linux-gnu/libc.so.6 (path dev=0,71)
bash         10921   dantopa   mem      REG   254,1    1190606  /usr/lib/x86_64-linux-gnu/libtinfo.so.6.3 (path dev=0,71)
bash         10921   dantopa   mem      REG   254,1    63095935  /usr/lib/x86_64-linux-gnu/ld-linux-x86-64.so.2 (path dev=0,71)
bash         10921   dantopa   0u       CHR   136,0      0t0      3  /dev/pts/0
bash         10921   dantopa   1u       CHR   136,0      0t0      3  /dev/pts/0
bash         10921   dantopa   2u       CHR   136,0      0t0      3  /dev/pts/0
bash         10921   dantopa   255u    CHR   136,0      0t0      3  /dev/pts/0
bash         10932   dantopa   cwd      DIR    0,33     704    1572  /repos/github/vault-fortran/Xmodern-fortran/
bash         10932   dantopa   rtd      DIR    0,71    4096  61653409  /
bash         10932   dantopa   txt      REG    0,71   1396520  62702252  /usr/bin/bash
bash         10932   dantopa   mem      REG   254,1    62702252  /usr/bin/bash (path dev=0,71)
bash         10932   dantopa   mem      REG   254,1    63095938  /usr/lib/x86_64-linux-gnu/libc.so.6 (path dev=0,71)
bash         10932   dantopa   mem      REG   254,1    1190606  /usr/lib/x86_64-linux-gnu/libtinfo.so.6.3 (path dev=0,71)
```

```

bash      10932 dantopa mem    REG  254,1      63095935 /usr/lib/x86_64-linux-gnu/ld-linux-x86-64.so
bash      10932 dantopa 0u    CHR  136,1      0t0      4 /dev/pts/1
bash      10932 dantopa 1u    CHR  136,1      0t0      4 /dev/pts/1
bash      10932 dantopa 2u    CHR  136,1      0t0      4 /dev/pts/1
bash      10932 dantopa 255u  CHR  136,1      0t0      4 /dev/pts/1
lsof      11139 dantopa cwd    DIR   0,33      704      1572 /repos/github/vault-fortran/Xmodern-fortran/
lsof      11139 dantopa rtd    DIR   0,71     4096 61653409 /
lsof      11139 dantopa txt    REG   0,71    167544 709329 /usr/bin/lsof
lsof      11139 dantopa mem    REG  254,1      709329 /usr/bin/lsof (path dev=0,71)
lsof      11139 dantopa mem    REG  254,1     63095951 /usr/lib/x86_64-linux-gnu/libresolv.so.2 (pa
lsof      11139 dantopa mem    REG  254,1     1190531 /usr/lib/x86_64-linux-gnu/libkeyutils.so.1.9
lsof      11139 dantopa mem    REG  254,1     63096020 /usr/lib/x86_64-linux-gnu/libkrb5support.so.
lsof      11139 dantopa mem    REG  254,1     63096026 /usr/lib/x86_64-linux-gnu/libcom_err.so.2.1
lsof      11139 dantopa mem    REG  254,1     63096018 /usr/lib/x86_64-linux-gnu/libk5crypto.so.3.1
lsof      11139 dantopa mem    REG  254,1     63096022 /usr/lib/x86_64-linux-gnu/libkrb5.so.3.3 (pa
lsof      11139 dantopa mem    REG  254,1     1190578 /usr/lib/x86_64-linux-gnu/libpcre2-8.so.0.10
lsof      11139 dantopa mem    REG  254,1     63096024 /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.
lsof      11139 dantopa mem    REG  254,1     63095938 /usr/lib/x86_64-linux-gnu/libc.so.6 (path de
lsof      11139 dantopa mem    REG  254,1     1190588 /usr/lib/x86_64-linux-gnu/libselinux.so.1 (p
lsof      11139 dantopa mem    REG  254,1     1190608 /usr/lib/x86_64-linux-gnu/libtirpc.so.3.0.0
lsof      11139 dantopa mem    REG  254,1     63095935 /usr/lib/x86_64-linux-gnu/ld-linux-x86-64.so
lsof      11139 dantopa 0u    CHR  136,1      0t0      4 /dev/pts/1
lsof      11139 dantopa 1u    CHR  136,1      0t0      4 /dev/pts/1
lsof      11139 dantopa 2u    CHR  136,1      0t0      4 /dev/pts/1
lsof      11139 dantopa 3r    DIR   0,74        0        1 /proc
lsof      11139 dantopa 4r    DIR   0,74        7    123326 /proc/11139/fd
lsof      11139 dantopa 5w    FIFO  0,11      0t0    123331 pipe
lsof      11139 dantopa 6r    FIFO  0,11      0t0    123332 pipe
lsof      11140 dantopa cwd    DIR   0,33      704      1572 /repos/github/vault-fortran/Xmodern-fortran/
lsof      11140 dantopa rtd    DIR   0,71     4096 61653409 /
lsof      11140 dantopa txt    REG   0,71    167544 709329 /usr/bin/lsof
lsof      11140 dantopa mem    REG  254,1      709329 /usr/bin/lsof (path dev=0,71)
lsof      11140 dantopa mem    REG  254,1     63095951 /usr/lib/x86_64-linux-gnu/libresolv.so.2 (pa
lsof      11140 dantopa mem    REG  254,1     1190531 /usr/lib/x86_64-linux-gnu/libkeyutils.so.1.9
lsof      11140 dantopa mem    REG  254,1     63096020 /usr/lib/x86_64-linux-gnu/libkrb5support.so.
lsof      11140 dantopa mem    REG  254,1     63096026 /usr/lib/x86_64-linux-gnu/libcom_err.so.2.1
lsof      11140 dantopa mem    REG  254,1     63096018 /usr/lib/x86_64-linux-gnu/libk5crypto.so.3.1
lsof      11140 dantopa mem    REG  254,1     63096022 /usr/lib/x86_64-linux-gnu/libkrb5.so.3.3 (pa
lsof      11140 dantopa mem    REG  254,1     1190578 /usr/lib/x86_64-linux-gnu/libpcre2-8.so.0.10
lsof      11140 dantopa mem    REG  254,1     63096024 /usr/lib/x86_64-linux-gnu/libgssapi_krb5.so.
lsof      11140 dantopa mem    REG  254,1     63095938 /usr/lib/x86_64-linux-gnu/libc.so.6 (path de
lsof      11140 dantopa mem    REG  254,1     1190588 /usr/lib/x86_64-linux-gnu/libselinux.so.1 (p
lsof      11140 dantopa mem    REG  254,1     1190608 /usr/lib/x86_64-linux-gnu/libtirpc.so.3.0.0
lsof      11140 dantopa mem    REG  254,1     63095935 /usr/lib/x86_64-linux-gnu/ld-linux-x86-64.so
lsof      11140 dantopa 4r    FIFO  0,11      0t0    123331 pipe
lsof      11140 dantopa 7w    FIFO  0,11      0t0    123332 pipe

```

2.5 objdump

The `objdump` command shows dependent shared objects, typically libraries. Two versions of the shared library for the GNU standard C library – one 32 bit, the other 64 bit – are located.

```

dantopa@92bc4c447e32:/$ locate libc.so.6
/usr/lib/x86_64-linux-gnu/libc.so.6
/usr/lib32/libc.so.6

```

2.6 readelf

The `readelf` displays information about ELF files, or Executable and Linkable Format files which are a standard file format for executable files, object code, shared libraries, and core dumps.¹ This example lists the header file for the command `bash`.

```
dantopa@92bc4c447e32:~$ file /bin/bash
/bin/bash: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /li
BuildID[sha1]=7a6408ba82a2d86dd98f1f75ac8edcb695f6fd60, for GNU/Linux 3.2.0, stripped
dantopa@92bc4c447e32:~$ readelf -h /bin/bash
ELF Header:
  Magic:   7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
  Class:                                ELF64
  Data:                                  2's complement, little endian
  Version:                              1 (current)
  OS/ABI:                                UNIX - System V
  ABI Version:                           0
  Type:                                  DYN (Position-Independent Executable file)
  Machine:                               Advanced Micro Devices X86-64
  Version:                               0x1
  Entry point address:                   0x32ef0
  Start of program headers:              64 (bytes into file)
  Start of section headers:              1394600 (bytes into file)
  Flags:                                 0x0
  Size of this header:                   64 (bytes)
  Size of program headers:               56 (bytes)
  Number of program headers:              13
  Size of section headers:               64 (bytes)
  Number of section headers:              30
  Section header string table index: 29
```

2.7 nm

The `nm` command shows dependent shared objects and executables;

2.8 strace

The `strace` command is very powerful and the following examples.

2.8.1 Trace System Calls To A Given Path

```
root@169e8b2c1ae3:/# strace -P /etc/ld.so.cache ls /dev/null
openat(AT_FDCWD, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3
newfstatat(3, "", st_mode=S_IFREG|0644, st_size=135191, ..., AT_EMPTY_PATH) = 0
mmap(NULL, 135191, PROT_READ, MAP_PRIVATE, 3, 0) = 0x7f03bba95000
close(3) = 0
/dev/null
+++ exited with 0 +++
```

2.8.2 Inventory time, calls, and errors for every system call

```
root@169e8b2c1ae3:/# strace -c ls > /dev/null
% time      seconds  usecs/call   calls   errors syscall
-----
 71.76      0.013546    6773         2         getdents64
  7.85      0.001482     247         6         openat
  4.88      0.000922     922         1         execve
```

¹For an ELF cheatsheet see <https://gist.github.com/x0nu11byt3/bcb35c3de461e5fb66173071a2379779>.

4.44	0.000839	49	17	mmap
1.84	0.000347	43	8	close
1.48	0.000279	39	7	mprotect
1.40	0.000265	37	7	newfstatat
1.26	0.000237	47	5	read
0.94	0.000178	44	4	pread64
0.77	0.000145	48	3	brk
0.57	0.000108	36	3	3 ioctl
0.49	0.000092	46	2	2 statfs
0.47	0.000088	44	2	2 access
0.34	0.000065	32	2	1 arch_prctl
0.34	0.000065	65	1	getrandom
0.32	0.000061	61	1	munmap
0.18	0.000034	34	1	rseq
0.17	0.000032	32	1	set_robust_list
0.16	0.000031	31	1	write
0.16	0.000031	31	1	set_tid_address
0.16	0.000031	31	1	prlimit64

100.00	0.018878	248	76	8 total

2.8.3 Identify Information Associated With File Descriptors

```

root@169e8b2c1ae3:/# strace -yy cat /dev/null
execve("/usr/bin/cat", ["cat", "/dev/null"], 0x7ffffb8b235d0 /* 10 vars */) = 0
brk(NULL)                                = 0x5611c6a38000
arch_prctl(0x3001 /* ARCH_??? */, 0x7ffeede990c0) = -1 EINVAL (Invalid argument)
mmap(NULL, 8192, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5c648b8000
access("/etc/ld.so.preload", R_OK)       = -1 ENOENT (No such file or directory)
openat(AT_FDCWD</>, "/etc/ld.so.cache", O_RDONLY|O_CLOEXEC) = 3</etc/ld.so.cache>
newfstatat(3</etc/ld.so.cache>, "", st_mode=S_IFREG|0644, st_size=135191, ..., AT_EMPTY_PATH) = 0
mmap(NULL, 135191, PROT_READ, MAP_PRIVATE, 3</etc/ld.so.cache>, 0) = 0x7f5c64896000
close(3</etc/ld.so.cache>)              = 0
openat(AT_FDCWD</>, "/lib/x86_64-linux-gnu/libc.so.6", O_RDONLY|O_CLOEXEC) = 3</usr/lib/x86_64-linux-gnu/
read(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "\ 177ELF\2\1\1\3\0\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P\237\2\0\
pread64(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "\6\0\0\0\4\0\0\0@0\0\0\0\0\0\0@0\0\0\0\0\0\0@0\0\0\0\
pread64(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "\4\0\0\0 \0\0\0\5\0\0\0GNU\0\2\0\0\300\4\0\0\0\3\0\0\0\
pread64(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "\4\0\0\0\24\0\0\0\3\0\0\0GNU\0I\17\357\204\3$\ f\221\2\
newfstatat(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "", st_mode=S_IFREG|0755, st_size=2220400, ..., AT_E
pread64(3</usr/lib/x86_64-linux-gnu/libc.so.6>, "\6\0\0\0\4\0\0\0@0\0\0\0\0\0\0@0\0\0\0\0\0\0@0\0\0\0\
mmap(NULL, 2264656, PROT_READ, MAP_PRIVATE|MAP_DENYWRITE, 3</usr/lib/x86_64-linux-gnu/libc.so.6>, 0) = 0
mprotect(0x7f5c64695000, 2023424, PROT_NONE) = 0
mmap(0x7f5c64695000, 1658880, PROT_READ|PROT_EXEC, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3</usr/lib/x86_6
mmap(0x7f5c6482a000, 360448, PROT_READ, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3</usr/lib/x86_64-linux-gn
mmap(0x7f5c64883000, 24576, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_DENYWRITE, 3</usr/lib/x86_6
mmap(0x7f5c64889000, 52816, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_FIXED|MAP_ANONYMOUS, -1, 0) = 0x7f5c6
close(3</usr/lib/x86_64-linux-gnu/libc.so.6>) = 0
mmap(NULL, 12288, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5c6466a000
arch_prctl(ARCH_SET_FS, 0x7f5c6466a740) = 0
set_tid_address(0x7f5c6466aa10)          = 23663
set_robust_list(0x7f5c6466aa20, 24)      = 0
rseq(0x7f5c6466b0e0, 0x20, 0, 0x53053053) = 0
mprotect(0x7f5c64883000, 16384, PROT_READ) = 0
mprotect(0x5611c4bde000, 4096, PROT_READ) = 0
mprotect(0x7f5c648f2000, 8192, PROT_READ) = 0
prlimit64(0, RLIMIT_STACK, NULL, rlim_cur=8192*1024, rlim_max=RLIM64_INFINITY) = 0
munmap(0x7f5c64896000, 135191)           = 0
getrandom("\ x7e\ x74\ x62\ xbc\ x66\ x05\ x81\ xf8", 8, GRND_NONBLOCK) = 8

```

```

brk(NULL) = 0x5611c6a38000
brk(0x5611c6a59000) = 0x5611c6a59000
newfstatat(1</dev/pts/0<char 136:0>>, "", st_mode=S_IFCHR|0620, st_rdev=makedev(0x88, 0), ..., AT_EMPTY_
openat(AT_FDCWD</>, "/dev/null", O_RDONLY) = 3</dev/null<char 1:3>>
newfstatat(3</dev/null<char 1:3>>, "", st_mode=S_IFCHR|0666, st_rdev=makedev(0x1, 0x3), ..., AT_EMPTY_P
fadvise64(3</dev/null<char 1:3>>, 0, 0, POSIX_FADV_SEQUENTIAL) = 0
mmap(NULL, 139264, PROT_READ|PROT_WRITE, MAP_PRIVATE|MAP_ANONYMOUS, -1, 0) = 0x7f5c64896000
read(3</dev/null<char 1:3>>, "", 131072) = 0
munmap(0x7f5c64896000, 139264) = 0
close(3</dev/null<char 1:3>>) = 0
close(1</dev/pts/0<char 136:0>>) = 0
close(2</dev/pts/0<char 136:0>>) = 0
exit_group(0) = ?
+++ exited with 0 +++

```

2.9 strings

Stub for strings.

3 Manual Pages

3.1 ldd: Print Shared Object Dependencies

NAME	ldd - print shared object dependencies
SYNOPSIS	ldd [option]... file...
DESCRIPTION	<p>ldd prints the shared objects (shared libraries) required by each program or shared object specified on the command line. An example of its use and output is the following:</p> <pre> \$ ldd /bin/ls linux-vdso.so.1 (0x00007ffcc3563000) libselinux.so.1 => /lib64/libselinux.so.1 (0x00007f87e5459000) libcap.so.2 => /lib64/libcap.so.2 (0x00007f87e5254000) libc.so.6 => /lib64/libc.so.6 (0x00007f87e4e92000) libpcre.so.1 => /lib64/libpcre.so.1 (0x00007f87e4c22000) libdl.so.2 => /lib64/libdl.so.2 (0x00007f87e4a1e000) /lib64/ld-linux-x86-64.so.2 (0x00005574b12e000) libattr.so.1 => /lib64/libattr.so.1 (0x00007f87e4817000) libpthread.so.0 => /lib64/libpthread.so.0 (0x00007f87e45fa000) </pre> <p>In the usual case, ldd invokes the standard dynamic linker (see ld.so(8)) with the LD_TRACE_LOADED_OBJECTS environment variable set to 1. This causes the dynamic linker to inspect the program's dynamic dependencies, and find (according to the rules described in ld.so(8)) and load the objects that satisfy those dependencies. For each dependency, ldd displays the location of the matching object and the (hexadecimal) address at which it is loaded. (The linux-vdso and ld-linux shared dependencies are special; see vdso(7) and ld.so(8).)</p>

Security

Be aware that **in** some circumstances (e.g., where the program specifies an ELF interpreter other than `ld-linux.so`), some versions of `ldd` may attempt to obtain the dependency information by attempting to directly execute the program, which may lead to the execution of whatever code is defined **in** the program's ELF interpreter, and perhaps to execution of the program itself. (Before `glibc 2.27`, the upstream `ldd` implementation did this **for** example, although most distributions provided a modified version that did not.)

Thus, you should never employ `ldd` on an untrusted executable, since this may result **in** the execution of arbitrary code. A safer alternative when dealing with untrusted executables is:

```
$ objdump -p /path/to/program | grep NEEDED
```

Note, however, that this alternative shows only the direct dependencies of the executable, **while** `ldd` shows the entire dependency tree of the executable.

OPTIONS

`--version`
Print the version number of `ldd`.

`--verbose`
`-v` Print all information, including, **for** example, symbol versioning information.

`--unused`
`-u` Print unused direct dependencies. (Since `glibc 2.3.4`.)

`--data-relocs`
`-d` Perform relocations and report any missing objects (ELF only).

`--function-relocs`
`-r` Perform relocations **for** both data objects and functions, and report any missing objects or functions (ELF only).

`--help` Usage information.

BUGS

`ldd` does not work on `a.out` shared libraries.

`ldd` does not work with some extremely old `a.out` programs which were built before `ldd` support was added to the compiler releases. If you use `ldd` on one of these programs, the program will attempt to run with `argc = 0` and the results will be unpredictable.

SEE ALSO

`pldd(1)`, `sprof(1)`, `ld.so(8)`, `ldconfig(8)`

COLOPHON

This page is part of the man-pages (Linux kernel and C library user-space interface documentation) project. Information about the project can be found at <https://www.kernel.org/doc/man-pages/>. If you have a bug report **for** this manual page, see <https://git.kernel.org/pub/scm/docs/man-pages/man-pages.git/tree/CONTRIBUTING>. This page was obtained from the tarball man-pages-6.9.1.tar.gz fetched from <https://mirrors.edge.kernel.org/pub/linux/docs/man-pages/> on 2024-06-26. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

Linux man-pages 6.9.1

2024-05-02

ldd(1)

3.2 lddconfig: Configure Dynamic Linker Run-time Bindings

NAME

ldconfig - configure dynamic linker run-time bindings

SYNOPSIS

/sbin/ldconfig [-nNvVX] [-C cache] [-f conf] [-r root]
directory ...

/sbin/ldconfig -l [-v] library ...

/sbin/ldconfig -p

DESCRIPTION

ldconfig creates the necessary links and cache to the most recent shared libraries found **in** the directories specified on the **command** line, **in** the file /etc/ld.so.conf, and **in** the trusted directories, /lib and /usr/lib. On some 64-bit architectures such as x86-64, /lib and /usr/lib are the trusted directories **for** 32-bit libraries, **while** /lib64 and /usr/lib64 are used **for** 64-bit libraries.

The cache is used by the run-time linker, ld.so or ld-linux.so. ldconfig checks the header and filenames of the libraries it encounters when determining which versions should have their links updated. ldconfig should normally be run by the superuser as it may require write permission on some root owned directories and files.

ldconfig will look only at files that are named lib*.so* (**for** regular shared objects) or ld-*.so* (**for** the dynamic loader itself). Other files will be ignored. Also, ldconfig expects a

certain pattern to how the symbolic links are **set** up, like this example, where the middle file (libfoo.so.1 here) is the SONAME **for** the library:

```
libfoo.so -> libfoo.so.1 -> libfoo.so.1.12
```

Failure to follow this pattern may result **in** compatibility issues after an upgrade.

OPTIONS

- format=fmt
- c fmt (Since glibc 2.2) Use cache format fmt, which is one of old, new, or compat. Since glibc 2.32, the default is new. Before that, it was compat.
- C cache
Use cache instead of /etc/ld.so.cache.
- f conf
Use conf instead of /etc/ld.so.conf.
- ignore-aux-cache
- i (Since glibc 2.7) Ignore auxiliary cache file.
- l (Since glibc 2.2) Interpret each operand as a library name and configure its links. Intended **for** use only by experts.
- n Process only the directories specified on the **command** line; don't process the trusted directories, nor those specified **in** /etc/ld.so.conf. Implies -N.
- N Don't rebuild the cache. Unless -X is also specified, links are still updated.
- print-cache
- p Print the lists of directories and candidate libraries stored **in** the current cache.
- r root
Change to and use root as the root directory.
- verbose
- v Verbose mode. Print current version number, the name of each directory as it is scanned, and any links that are created. Overrides quiet mode.
- version
- V Print program version.
- X Don't update links. Unless -N is also specified, the

cache is still rebuilt.

FILES

/lib/ld.so
is the run-time linker/loader.
/etc/ld.so.conf
contains a list of directories, one per line, **in** which to search **for** libraries.
/etc/ld.so.cache
contains an ordered list of libraries found **in** the directories specified **in** /etc/ld.so.conf, as well as those found **in** the trusted directories.

SEE ALSO

ldd(1), ld.so(8)

COLOPHON

This page is part of the man-pages (Linux kernel and C library user-space interface documentation) project. Information about the project can be found at <https://www.kernel.org/doc/man-pages/>. If you have a bug report **for** this manual page, see <https://git.kernel.org/pub/scm/docs/man-pages/man-pages.git/tree/CONTRIBUTING>. This page was obtained from the tarball man-pages-6.9.1.tar.gz fetched from <https://mirrors.edge.kernel.org/pub/linux/docs/man-pages/> on 2024-06-26. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

Linux man-pages 6.9.1

2024-05-02

ldconfig(8)

3.3 locate: List File in Databases

NAME

locate - list files **in** databases that match a pattern

SYNOPSIS

locate [-d path | --database=path] [-e | -E | --[non-]existing]
[-i | --ignore-**case**] [-0 | --null] [-c | --count] [-w |
--wholename] [-b | --basename] [-l N | --limit=N] [-S |
--statistics] [-r | --regex] [--regextype R] [--max-database-age
D] [-P | -H | --nofollow] [-L | --follow] [--version] [-A |
--all] [-p | --print] [--**help**] pattern...

DESCRIPTION

This manual page documents the GNU version of locate. For each given pattern, locate searches one or more databases of file names and displays the file names that contain the pattern. Patterns can contain shell-style metacharacters: `*', `?', and `[]'. The metacharacters **do** not treat `/' or `.' specially.

Therefore, a pattern ``foo*bar'` can match a file name that contains ``foo3/bar'`, and a pattern ``*duck*'` can match a file name that contains ``lake/.ducky'`. Patterns that contain metacharacters should be quoted to protect them from expansion by the shell.

If a pattern is a plain string - it contains no metacharacters - locate displays all file names **in** the database that contain that string anywhere. If a pattern does contain metacharacters, locate only displays file names that match the pattern exactly. As a result, patterns that contain metacharacters should usually begin with a ``*'``, and will most often end with one as well. The exceptions are patterns that are intended to explicitly match the beginning or end of a file name.

The file name databases contain lists of files that were on the system when the databases were last updated. The system administrator can choose the file name of the default database, the frequency with which the databases are updated, and the directories **for** which they contain entries; see `updatedb(1)`.

If locate's output is going to a terminal, unusual characters **in** the output are escaped **in** the same way as **for** the `-print` action of the `find` command. If the output is not going to a terminal, file names are printed exactly as-is.

OPTIONS

- `-0, --null`
Use ASCII NUL as a separator, instead of newline.
- `-A, --all`
Print only names which match all non-option arguments, not those matching one or more non-option arguments.
- `-b, --basename`
Results are considered to match **if** the pattern specified matches the final component of the name of a file as listed **in** the database. This final component is usually referred to as the ``base name'`.
- `-c, --count`
Instead of printing the matched filenames, just print the total number of matches we found, unless `--print (-p)` is also present.
- `-d path, --database=path`
Instead of searching the default file name database, search the file name databases **in** path, which is a colon-separated list of database file names. You can also use the environment variable `LOCATE_PATH` to **set** the list of database files to search. The option overrides the

environment variable **if** both are used. Empty elements **in** the path are taken to be synonyms **for** the file name of the default database. A database can be supplied on stdin, using `^-` as an element of path. If more than one element of path is `^-`, later instances are ignored (and a warning message is printed).

The file name database format changed starting with GNU find and locate version 4.0 to allow machines with different byte orderings to share the databases. This version of locate can automatically recognize and **read** databases produced **for** older versions of GNU locate or Unix versions of locate or find. Support **for** the old locate database format will be discontinued **in** a future release.

-e, --existing

Only print out such names that currently exist (instead of such names that existed when the database was created). Note that this may slow down the program a lot, **if** there are many matches **in** the database. If you are using this option within a program, please note that it is possible **for** the file to be deleted after locate has checked that it exists, but before you use it.

-E, --non-existing

Only print out such names that currently **do** not exist (instead of such names that existed when the database was created). Note that this may slow down the program a lot, **if** there are many matches **in** the database.

--help Print a summary of the options to locate and exit.

-i, --ignore-case

Ignore **case** distinctions **in** both the pattern and the file names.

-l N, --limit=N

Limit the number of matches to N. If a limit is **set** via this option, the number of results printed **for** the **-c** option will never be larger than this number.

-L, --follow

If testing **for** the existence of files (with the **-e** or **-E** options), consider broken symbolic links to be non-existing. This is the default.

--max-database-age D

Normally, locate will issue a warning message when it searches a database which is more than 8 days old. This

option changes that value to something other than 8. The effect of specifying a negative value is undefined.

-m, --mmap

Accepted but does nothing, **for** compatibility with BSD locate.

-P, -H, --nofollow

If testing **for** the existence of files (with the -e or -E options), treat broken symbolic links as **if** they were existing files. The -H form of this option is provided purely **for** similarity with find; the use of -P is recommended over -H.

-p, --print

Print search results when they normally would not, because of the presence of --statistics (-S) or --count (-c).

-r, --regex

The pattern specified on the **command** line is understood to be a regular expression, as opposed to a glob pattern. The Regular expressions work **in** the same way as **in** emacs except **for** the fact that "." will match a newline. GNU find uses the same regular expressions. Filenames whose full paths match the specified regular expression are printed (or, **in** the **case** of the -c option, counted). If you wish to anchor your regular expression at the ends of the full path name, **then** as is usual with regular expressions, you should use the characters ^ and \$ to signify this.

--regextype R

Use regular expression dialect R. Supported dialects include 'findutils-default', 'posix-awk', 'posix-basic', 'posix-egrep', 'posix-extended', 'posix-minimal-basic', 'awk', 'ed', 'egrep', 'emacs', 'gnu-awk', 'grep' and 'sed'. See the Texinfo documentation **for** a detailed explanation of these dialects.

-s, --stdio

Accepted but does nothing, **for** compatibility with BSD locate.

-S, --statistics

Print various statistics about each locate database and **then exit** without performing a search, unless non-option arguments are given. For compatibility with BSD, -S is accepted as a synonym **for** --statistics. However, the output of locate -S is different **for** the GNU and BSD implementations of locate.

`--version`
Print the version number of locate and exit.

`-w, --wholename`
Match against the whole name of the file as listed **in** the database. This is the default.

ENVIRONMENT

`LOCATE_PATH`
Colon-separated list of databases to search. If the value has a leading or trailing colon, or has two colons **in** a row, you may get results that vary between different versions of locate.

HISTORY

The locate program started life as the BSD fast find program, contributed to BSD by James A. Woods. This was described by his paper Finding Files Fast which was published **in** Usenix ;**login**:, Vol 8, No 1, February/March, 1983, pp. 8-10. When the find program began to assume a default `-print` action **if** no action was specified, this changed the interpretation of find pattern. The BSD developers therefore moved the fast find functionality into locate. The GNU implementation of locate appears to be derived from the same code.

Significant changes to locate **in** reverse order:

4.3.7 Byte-order independent support **for** old database format
4.3.3 `locate -i` supports multi-byte characters correctly
Introduced `--max_db_age`
4.3.2 Support **for** the `slocate` database format
4.2.22 Introduced the `--all` option
4.2.15 Introduced the `--regex` option
4.2.14 Introduced options `-L`, `-P`, `-H`
4.2.12 Empty items **in** `LOCATE_PATH` now indicate the default database
4.2.11 Introduced the `--statistics` option
4.2.4 Introduced `--count` and `--limit`
4.2.0 Glob characters cause matching against the whole file name
4.0 Introduced the `LOCATE02` database format
3.7 Locate can search multiple databases

BUGS

The locate database correctly handles filenames containing newlines, but only **if** the system's sort **command** has a working `-z` option. If you suspect that locate may need to **return** filenames containing newlines, consider using its `--null` option.

REPORTING BUGS

GNU findutils online **help**:
<<https://www.gnu.org/software/findutils/#get-help>>
Report any translation bugs to
<<https://translationproject.org/team/>>

Report any other issue via the form at the GNU Savannah bug

tracker:
 <<https://savannah.gnu.org/bugs/?group=findutils>>
General topics about the GNU findutils package are discussed at
the bug-findutils mailing list:
 <<https://lists.gnu.org/mailman/listinfo/bug-findutils>>

COPYRIGHT

Copyright (C) 1994-2024 Free Software Foundation, Inc. License
GPLv3+: GNU GPL version 3 or later
<<https://gnu.org/licenses/gpl.html>>.
This is free software: you are free to change and redistribute
it. There is NO WARRANTY, to the extent permitted by law.

SEE ALSO

find(1), updatedb(1), xargs(1), glob(3), locatedb(5)

Full documentation
<<https://www.gnu.org/software/findutils/locate>>
or available locally via: info locate

COLOPHON

This page is part of the findutils (find utilities) project.
Information about the project can be found at
<http://www.gnu.org/software/findutils/>. If you have a bug
report **for** this manual page, see
<https://savannah.gnu.org/bugs/?group=findutils>. This page was
obtained from the project's upstream Git repository
[git://git.savannah.gnu.org/findutils.git](https://git.savannah.gnu.org/findutils.git) on 2024-06-14. (At
that time, the date of the most recent commit that was found **in**
the repository was 2024-06-03.) If you discover any rendering
problems **in** this HTML version of the page, or you believe there
is a better or more up-to-date **source for** the page, or you have
corrections or improvements to the information **in** this COLOPHON
(which is not part of the original manual page), send a mail to
man-pages@man7.org

LOCATE(1)

3.4 lsof: Show Open Files

NAME

lsof - list open files

SYNOPSIS

```
lsof [ -?abChlnNOPRtUvVX ] [ -A A ] [ -c c ] [ +c c ] [ +|-d d ]  
[ +|-D D ] [ +|-e s ] [ +|-E ] [ +|-f [cfgGn] ] [ -F [f] ] [ -g  
[s] ] [ -i [i] ] [ -k k ] [ -K k ] [ +|-L [l] ] [ +|-m m ] [ +|-M  
] [ -o [o] ] [ -p s ] [ +|-r [t[m<fmt>]] ] [ -s [p:s] ] [ -S [t]  
] [ -T [t] ] [ -u s ] [ +|-w ] [ -x [fl] ] [ -z [z] ] [ -Z [Z] ]  
[ -- ] [names]
```

DESCRIPTION

Lsof revision 4.91 lists on its standard output file information
about files opened by processes **for** the following UNIX dialects:

Apple Darwin 9 and Mac OS X 10.[567]
FreeBSD 8.[234], 9.0 and 1[012].0 **for** AMD64-based systems
Linux 2.1.72 and above **for** x86-based systems
Solaris 9, 10 and 11

(See the DISTRIBUTION section of this manual page **for** information on how to obtain the latest lsof revision.)

An open file may be a regular file, a directory, a block special file, a character special file, an executing text reference, a library, a stream or a network file (Internet socket, NFS file or UNIX domain socket.) A specific file or all the files **in** a file system may be selected by path.

Instead of a formatted display, lsof will produce output that can be parsed by other programs. See the -F, option description, and the OUTPUT FOR OTHER PROGRAMS section **for** more information.

In addition to producing a single output list, lsof will run **in** repeat mode. In repeat mode it will produce output, delay, **then** repeat the output operation **until** stopped with an interrupt or quit signal. See the +|-r [t[m<fmt>]] option description **for** more information.

OPTIONS

In the absence of any options, lsof lists all open files belonging to all active processes.

If any list request option is specified, other list requests must be specifically requested - e.g., **if** -U is specified **for** the listing of UNIX socket files, NFS files won't be listed unless -N is also specified; or **if** a user list is specified with the -u option, UNIX domain socket files, belonging to users not **in** the list, won't be listed unless the -U option is also specified.

Normally list options that are specifically stated are ORed - i.e., specifying the -i option without an address and the -ufoo option produces a listing of all network files OR files belonging to processes owned by user ``foo''. The exceptions are:

- 1) the `^' (negated) **login** name or user ID (UID), specified with the -u option;
- 2) the `^' (negated) process ID (PID), specified with the -p option;
- 3) the `^' (negated) process group ID (PGID), specified with the -g option;
- 4) the `^' (negated) **command**, specified with the -c option;

5) the (^) negated TCP or UDP protocol state names, specified with the -s [p:s] option.

Since they represent exclusions, they are applied without ORing or ANDing and take effect before any other selection criteria are applied.

The -a option may be used to AND the selections. For example, specifying -a, -U, and -ufoo produces a listing of only UNIX socket files that belong to processes owned by user ``foo''.

Caution: the -a option causes all list selection options to be ANDed; it can't be used to cause ANDing of selected pairs of selection options by placing it between them, even though its placement there is acceptable. Wherever -a is placed, it causes the ANDing of all selection options.

Items of the same selection **set** - **command** names, file descriptors, network addresses, process identifiers, user identifiers, zone names, security contexts - are joined **in** a single ORed **set** and applied before the result participates **in** ANDing. Thus, **for** example, specifying -i@aaa.bbb, -i@ccc.ddd, -a, and -ufff,ggg will **select** the listing of files that belong to either **login** ``ffff'' OR ``ggg'' AND have network connections to either host aaa.bbb OR ccc.ddd.

Options may be grouped together following a single prefix -- e.g., the option **set** ``-a -b -C'' may be stated as -abC. However, since values are optional following +|-f, -F, -g, -i, +|-L, -o, +|-r, -s, -S, -T, -x and -z. when you have no values **for** them be careful that the following character isn't ambiguous. For example, -Fn might represent the -F and -n options, or it might represent the n field identifier character following the -F option. When ambiguity is possible, start a new option with a '-' character - e.g., ``-F -n''. If the next option is a file name, follow the possibly ambiguous option with ``--'' - e.g., ``-F -- name''.

Either the '+' or the '-' prefix may be applied to a group of options. Options that don't take on separate meanings **for** each prefix - e.g., -i - may be grouped under either prefix. Thus, **for** example, ``+M -i'' may be stated as ``+Mi'' and the group means the same as the separate options. Be careful of prefix grouping when one or more options **in** the group does take on separate meanings under different prefixes - e.g., +|-M; ``-iM'' is not the same request as ``-i +M''. When **in** doubt, use separate options with appropriate prefixes.

-? -h These two equivalent options **select** a usage (**help**) output

list. Lsof displays a shortened form of this output when it detects an error **in** the options supplied to it, after it has displayed messages explaining each error. (Escape the `?' character as your shell requires.)

- a causes list selection options to be ANDed, as described above.
- A A is available on systems configured **for** AFS whose AFS kernel code is implemented via dynamic modules. It allows the lsof user to specify A as an alternate name list file where the kernel addresses of the dynamic modules might be found. See the lsof FAQ (The FAQ section gives its location.) **for** more information about dynamic modules, their symbols, and how they affect lsof.
- b causes lsof to avoid kernel functions that might block - lstat(2), readlink(2), and stat(2).

See the BLOCKS AND TIMEOUTS and AVOIDING KERNEL BLOCKS sections **for** information on using this option.

- c c selects the listing of files **for** processes executing the **command** that begins with the characters of c. Multiple commands may be specified, using multiple -c options. They are joined **in** a single ORed **set** before participating **in** AND option selection.

If c begins with a `^', **then** the following characters specify a **command** name whose processes are to be ignored (excluded.)

If c begins and ends with a slash ('/'), the characters between the slashes are interpreted as a regular expression. Shell meta-characters **in** the regular expression must be quoted to prevent their interpretation by the shell. The closing slash may be followed by these modifiers:

- b the regular expression is a basic one.
- i ignore the **case** of letters.
- x the regular expression is an extended one (default).

See the lsof FAQ (The FAQ section gives its location.) **for** more information on basic and extended regular expressions.

The simple **command** specification is tested first. If that **test** fails, the **command** regular expression is applied. If

the simple **command test** succeeds, the **command** regular expression **test** isn't made. This may result in ``no **command** found for regex:'' messages when lsof's -V option is specified.

+c w defines the maximum number of initial characters of the name, supplied by the UNIX dialect, of the UNIX **command** associated with a process to be printed in the COMMAND column. (The lsof default is nine.)

Note that many UNIX dialects **do** not supply all **command** name characters to lsof in the files and structures from which lsof obtains **command** name. Often dialects limit the number of characters supplied in those sources. For example, Linux 2.4.27 and Solaris 9 both limit **command** name length to 16 characters.

If w is zero ('0'), all **command** characters supplied to lsof by the UNIX dialect will be printed.

If w is less than the length of the column title, ``COMMAND'', it will be raised to that length.

-C disables the reporting of any path name components from the kernel's name cache. See the KERNEL NAME CACHE section for more information.

+d s causes lsof to search for all open instances of directory s and the files and directories it contains at its top level. +d does NOT descend the directory tree, rooted at s. The +D D option may be used to request a full-descent directory tree search, rooted at directory D.

Processing of the +d option does not follow symbolic links within s unless the -x or -x l option is also specified. Nor does it search for open files on file system mount points on subdirectories of s unless the -x or -x f option is also specified.

Note: the authority of the user of this option limits it to searching for files that the user has permission to examine with the system stat(2) function.

-d s specifies a list of file descriptors (FDs) to exclude from or include in the output listing. The file descriptors are specified in the comma-separated set s - e.g., ``cwd,1,3'', ``^6,^2''. (There should be no spaces in the set.)

The list is an exclusion list if all entries of the set

begin with `^'. It is an inclusion list **if** no entry begins with `^'. Mixed lists are not permitted.

A file descriptor number range may be **in** the **set** as long as neither member is empty, both members are numbers, and the ending member is larger than the starting one - e.g., ``0-7'' or ``3-10''. Ranges may be specified **for** exclusion **if** they have the `^' prefix - e.g., ``^0-7'' excludes all file descriptors 0 through 7.

Multiple file descriptor numbers are joined **in** a single ORed **set** before participating **in** AND option selection.

When there are exclusion and inclusion members **in** the **set**, **lsf** reports them as errors and exits with a non-zero **return** code.

See the description of File Descriptor (FD) output values **in** the OUTPUT section **for** more information on file descriptor names.

+D D causes **lsf** to search **for** all open instances of directory D and all the files and directories it contains to its **complete** depth.

Processing of the +D option does not follow symbolic links within D unless the -x or -x l option is also specified. Nor does it search **for** open files on file system mount points on subdirectories of D unless the -x or -x f option is also specified.

Note: the authority of the user of this option limits it to searching **for** files that the user has permission to examine with the system **stat(2)** function.

Further note: **lsf** may process this option slowly and require a large amount of dynamic memory to **do** it. This is because it must descend the entire directory tree, rooted at D, calling **stat(2)** **for** each file and directory, building a list of all the files it finds, and searching that list **for** a match with every open file. When directory D is large, these steps can take a long time, so use this option prudently.

-D D directs **lsf**'s use of the device cache file. The use of this option is sometimes restricted. See the DEVICE CACHE FILE section and the sections that follow it **for** more information on this option.

-D must be followed by a **function** letter; the **function**

letter may optionally be followed by a path name. Lsof recognizes these **function** letters:

- ? - report device cache file paths
- b - build the device cache file
- i - ignore the device cache file
- r - **read** the device cache file
- u - **read** and update the device cache file

The b, r, and u functions, accompanied by a path name, are sometimes restricted. When these functions are restricted, they will not appear **in** the description of the -D option that accompanies -h or -? option output. See the DEVICE CACHE FILE section and the sections that follow it **for** more information on these functions and when they're restricted.

The ? **function** reports the **read**-only and write paths that lsof can use **for** the device cache file, the names of any environment variables whose values lsof will examine when forming the device cache file path, and the format **for** the personal device cache file path. (Escape the `?' character as your shell requires.)

When available, the b, r, and u functions may be followed by the device cache file's path. The standard default is .lsof_hostname **in** the home directory of the real user ID that executes lsof, but this could have been changed when lsof was configured and compiled. (The output of the -h and -? options show the current default prefix - e.g., ``.lsof''.) The suffix, hostname, is the first component of the host's name returned by gethostname(2).

When available, the b **function** directs lsof to build a new device cache file at the default or specified path.

The i **function** directs lsof to ignore the default device cache file and obtain its information about devices via direct calls to the kernel.

The r **function** directs lsof to **read** the device cache at the default or specified path, but prevents it from creating a new device cache file when none exists or the existing one is improperly structured. The r **function**, when specified without a path name, prevents lsof from updating an incorrect or outdated device cache file, or creating a new one **in** its place. The r **function** is always available when it is specified without a path name argument; it may be restricted by the permissions of the lsof process.

When available, the **u function** directs **ls** to **read** the device cache file at the default or specified path, **if** possible, and to rebuild it, **if** necessary. This is the default device cache file **function** when no **-D** option has been specified.

+|-e s exempts the file system whose path name is **s** from being subjected to kernel **function** calls that might block. The **+e** option exempts **stat(2)**, **lstat(2)** and most **readlink(2)** kernel **function** calls. The **-e** option exempts only **stat(2)** and **lstat(2)** kernel **function** calls. Multiple file systems may be specified with separate **+|-e** specifications and each may have **readlink(2)** calls exempted or not.

This option is currently implemented only **for** Linux.

CAUTION: this option can easily be mis-applied to other than the file system of interest, because it uses path name rather than the more reliable device and inode numbers. (Device and inode numbers are acquired via the potentially blocking **stat(2)** kernel call and are thus not available, but see the **+|-m m** option as a possible alternative way to supply device numbers.) Use this option with great care and fully specify the path name of the file system to be exempted.

When open files on exempted file systems are reported, it may not be possible to obtain all their information. Therefore, some information columns will be blank, the characters **``UNKN''** preface the values **in** the **TYPE** column, and the applicable exemption option is added **in** parentheses to the end of the **NAME** column. (Some device number information might be made available via the **+|-m m** option.)

+|-E **+E** specifies that Linux pipe, Linux UNIX socket and Linux pseudoterminal files should be displayed with endpoint information and the files of the endpoints should also be displayed. Note: UNIX socket file endpoint information is only available when the compile flags line of **-v** output contains **HASUXSOCKET**, and pseudoterminal endpoint information is only available when the compile flags line contains **HASPTYPT**.

Pipe endpoint information is displayed **in** the **NAME** column **in** the form **``PID,cmd,FDmode''**, where **PID** is the endpoint process ID; **cmd** is the endpoint process **command**; **FD** is the endpoint file's descriptor; and **mode** is the endpoint file's access mode.

Pseudoterminal endpoint information is displayed **in** the NAME column as ``->/dev/ptsmin PID,cmd,FDmode'' or ``PID,cmd,FDmode''. The first form is **for** a master device; the second, **for** a slave device. min is a slave device's minor device number; and PID, cmd, FD and mode are the same as with pipe endpoint information. Note: pseudoterminal endpoint information is only available when the compile flags line of -V output contains HASPTYEPT.

UNIX socket file endpoint information is displayed **in** the NAME column **in** the form ``**type**=TYPE ->INO=INODE PID,cmd,FDmode'', where TYPE is the socket **type**; INODE is the i-node number of the connected socket; and PID, cmd, FD and mode are the same as with pipe endpoint information. Note: UNIX socket file endpoint information is available only when the compile flags line of -v output contains HASUXSOCKEPT.

Multiple occurrences of this information can appear **in** a file's NAME column.

-E specifies that Linux pipe and Linux UNIX socket files should be displayed with endpoint information, but not the files of the endpoints.

+|-f [cfgGn]

f by itself clarifies how path name arguments are to be interpreted. When followed by c, f, g, G, or n **in** any combination it specifies that the listing of kernel file structure information is to be enabled ('+') or inhibited ('-').

Normally a path name argument is taken to be a file system name **if** it matches a mounted-on directory name reported by mount(8), or **if** it represents a block device, named **in** the mount output and associated with a mounted directory name. When +f is specified, all path name arguments will be taken to be file system names, and lsof will complain **if** any are not. This can be useful, **for** example, when the file system name (mounted-on device) isn't a block device. This happens **for** some CD-ROM file systems.

When -f is specified by itself, all path name arguments will be taken to be simple files. Thus, **for** example, the ``-f -- '/' arguments direct lsof to search **for** open files with a '/' path name, not all open files **in** the '/' (root) file system.

Be careful to make sure +f and -f are properly terminated

and aren't followed by a character (e.g., of the file or file system name) that might be taken as a parameter. For example, use ``--' after +f and -f as **in** these examples.

```
$ lsof +f -- /file/system/name
$ lsof -f -- /file/name
```

The listing of information from kernel file structures, requested with the +f [cfgGn] option form, is normally inhibited, and is not available **in** whole or part **for** some dialects - e.g., /proc-based Linux kernels below 2.6.22. When the prefix to f is a plus sign (`+'), these characters request file structure information:

c	file structure use count (not Linux)
f	file structure address (not Linux)
g	file flag abbreviations (Linux 2.6.22 and up)
G	file flags in hexadecimal (Linux 2.6.22 and up)
n	file structure node address (not Linux)

When the prefix is minus (`-') the same characters disable the listing of the indicated values.

File structure addresses, use counts, flags, and node addresses may be used to detect more readily identical files inherited by child processes and identical files **in** use by different processes. Lsof column output can be sorted by output columns holding the values and listed to identify identical file use, or lsof field output can be parsed by an AWK or Perl post-filter script, or by a C program.

-F f specifies a character list, f, that selects the fields to be output **for** processing by another program, and the character that terminates each output field. Each field to be output is specified with a single character **in** f. The field terminator defaults to NL, but may be changed to NUL (000). See the OUTPUT FOR OTHER PROGRAMS section **for** a description of the field identification characters and the field output process.

When the field selection character list is empty, all standard fields are selected (except the raw device field, security context and zone field **for** compatibility reasons) and the NL field terminator is used.

When the field selection character list contains only a zero (`0'), all fields are selected (except the raw device field **for** compatibility reasons) and the NUL terminator character is used.

Other combinations of fields and their associated field terminator character must be **set** with explicit entries **in** **f**, as described **in** the OUTPUT FOR OTHER PROGRAMS section.

When a field selection character identifies an item **ls**of does not normally list - e.g., PPID, selected with **-R** - specification of the field character - e.g., **``-FR''** - also selects the listing of the item.

When the field selection character list contains the single character **`?**, **ls**of will display a **help** list of the field identification characters. (Escape the **`?** character as your shell requires.)

-g [s] excludes or selects the listing of files **for** the processes whose optional process group IDentification (PGID) numbers are **in** the comma-separated **set s** - e.g., **``123''** or **``123,^456''**. (There should be no spaces **in** the set.)

PGID numbers that begin with **``^'** (negation) represent exclusions.

Multiple PGID numbers are joined **in** a single ORed **set** before participating **in** AND option selection. However, PGID exclusions are applied without ORing or ANDing and take effect before other selection criteria are applied.

The **-g** option also enables the output display of PGID numbers. When specified without a PGID **set** that's all it does.

-i [i] selects the listing of files any of whose Internet address matches the address specified **in i**. If no address is specified, this option selects the listing of all Internet and x.25 (HP-UX) network files.

If **-i4** or **-i6** is specified with no following address, only files of the indicated IP version, IPv4 or IPv6, are displayed. (An IPv6 specification may be used only **if** the dialects supports IPv6, as indicated by **``[46]''** and **``IPv[46]''** **in** **ls**of's **-h** or **-?** output.) Sequentially specifying **-i4**, followed by **-i6** is the same as specifying **-i**, and vice-versa. Specifying **-i4**, or **-i6** after **-i** is the same as specifying **-i4** or **-i6** by itself.

Multiple addresses (up to a limit of 100) may be specified with multiple **-i** options. (A port number or service name range is counted as one address.) They are joined **in** a single ORed **set** before participating **in** AND option

selection.

An Internet address is specified **in** the form (Items **in** square brackets are optional.):

[46][protocol][@hostname|hostaddr][:service|port]

where:

- 46 specifies the IP version, IPv4 or IPv6 that applies to the following address.
 - '6' may be specified only **if** the UNIX dialect supports IPv6. If neither '4' nor '6' is specified, the following address applies to all IP versions.
- protocol is a protocol name - TCP, UDP
- hostname is an Internet host name. Unless a specific IP version is specified, open network files associated with host names of all versions will be selected.
- hostaddr is a numeric Internet IPv4 address **in** dot form; or an IPv6 numeric address **in** colon form, enclosed **in** brackets, **if** the UNIX dialect supports IPv6. When an IP version is selected, only its numeric addresses may be specified.
- service is an /etc/services name - e.g., smtp - or a list of them.
- port is a port number, or a list of them.

IPv6 options may be used only **if** the UNIX dialect supports IPv6. To see **if** the dialect supports IPv6, run `lsof` and specify the `-h` or `-? (help)` option. If the displayed description of the `-i` option contains ```[46]''` and ```IPv[46]''`, IPv6 is supported.

IPv4 host names and addresses may not be specified **if** network file selection is limited to IPv6 with `-i 6`. IPv6 host names and addresses may not be specified **if** network file selection is limited to IPv4 with `-i 4`. When an open IPv4 network file's address is mapped **in** an IPv6 address, the open file's **type** will be IPv6, not IPv4, and its display will be selected by '6', not '4'.

At least one address component - 4, 6, protocol, hostname, hostaddr, or service - must be supplied. The ``@'` character, leading the host specification, is always required; as is the ``:'`, leading the port specification. Specify either hostname or hostaddr. Specify either service name list or port number list. If a service name list is specified, the protocol may also need to be

specified **if** the TCP, UDP and UDPLITE port numbers **for** the service name are different. Use any **case** - lower or upper - **for** protocol.

Service names and port numbers may be combined **in** a list whose entries are separated by commas and whose numeric range entries are separated by minus signs. There may be no embedded spaces, and all service names must belong to the specified protocol. Since service names may contain embedded minus signs, the starting entry of a range can't be a service name; it can be a port number, however.

Here are some sample addresses:

```
-i6 - IPv6 only
TCP:25 - TCP and port 25
@1.2.3.4 - Internet IPv4 host address 1.2.3.4
@[3ffe:1ebc::1]:1234 - Internet IPv6 host address
    3ffe:1ebc::1, port 1234
UDP:who - UDP who service port
TCP@lsof.itap:513 - TCP, port 513 and host name lsof.itap
tcp@foo:1-10,smtp,99 - TCP, ports 1 through 10,
    service name smtp, port 99, host name foo
tcp@bar:1-smtp - TCP, ports 1 through smtp, host bar
:time - either TCP, UDP or UDPLITE time service port
```

-K k selects the listing of tasks (threads) of processes, on dialects where task (thread) reporting is supported. (If **help** output - i.e., the output of the -h or -? options - shows this option, **then** task (thread) reporting is supported by the dialect.)

If -K is followed by a value, k, it must be `~i`. That causes lsof to ignore tasks, particularly **in** the default, list-everything **case** when no other options are specified.

When -K and -a are both specified on Linux, and the tasks of a main process are selected by other options, the main process will also be listed as though it were a task, but without a task ID. (See the description of the TID column **in** the OUTPUT section.)

Where the FreeBSD version supports threads, all threads will be listed with their IDs.

In general threads and tasks inherit the files of the **caller**, but may close some and open others, so lsof always reports all the open files of threads and tasks.

-k k specifies a kernel name list file, k, **in** place of /vmunix,

/mach, etc. -k is not available under AIX on the IBM RISC/System 6000.

-l inhibits the conversion of user ID numbers to **login** names. It is also useful when **login** name lookup is working improperly or slowly.

+|-L [l]

enables ('+') or disables ('-') the listing of file link counts, where they are available - e.g., they aren't available **for** sockets, or most FIFOs and pipes.

When +L is specified without a following number, all link counts will be listed. When -L is specified (the default), no link counts will be listed.

When +L is followed by a number, only files having a link count less than that number will be listed. (No number may follow -L.) A specification of the form ``+L1'' will **select** open files that have been unlinked. A specification of the form ``+a1 <file_system>'' will **select** unlinked open files on the specified file system.

For other link count comparisons, use field output (-F) and a post-processing script or program.

+|-m m specifies an alternate kernel memory file or activates mount table supplement processing.

The option form -m m specifies a kernel memory file, m, **in** place of /dev/kmem or /dev/mem - e.g., a crash dump file.

The option form +m requests that a mount supplement file be written to the standard output file. All other options are silently ignored.

There will be a line **in** the mount supplement file **for** each mounted file system, containing the mounted file system directory, followed by a single space, followed by the device number **in** hexadecimal "0x" format - e.g.,

/ 0x801

Lsof can use the mount supplement file to get device numbers **for** file systems when it can't get them via stat(2) or lstat(2).

The option form +m m identifies m as a mount supplement file.

Note: the `+m` and `+m m` options are not available **for** all supported dialects. Check the output of `lsnf's -h` or `-?` options to see **if** the `+m` and `+m m` options are available.

`+|-M` Enables (+) or disables (-) the reporting of portmapper registrations **for local** TCP, UDP and UDPLITE ports, where port mapping is supported. (See the last paragraph of this option description **for** information about where portmapper registration reporting is supported.)

The default reporting mode is **set** by the `lsnf` builder with the `HASPMAPENABLED` *#define* in the dialect's machine.h header file; `lsnf` is distributed with the `HASPMAPENABLED` *#define* deactivated, so portmapper reporting is disabled by default and must be requested with `+M`. Specifying `lsnf's -h` or `-?` option will report the default mode. Disabling portmapper registration when it is already disabled or enabling it when already enabled is acceptable. When portmapper registration reporting is enabled, `lsnf` displays the portmapper registration (**if** any) **for local** TCP, UDP or UDPLITE ports **in** square brackets immediately following the port numbers or service names - e.g., ```:1234[name]''` or ```:name[100083]''`. The registration information may be a name or number, depending on what the registering program supplied to the portmapper when it registered the port.

When portmapper registration reporting is enabled, `lsnf` may run a little more slowly or even become blocked when access to the portmapper becomes congested or stopped. Reverse the reporting mode to determine **if** portmapper registration reporting is slowing or blocking `lsnf`.

For purposes of portmapper registration reporting `lsnf` considers a TCP, UDP or UDPLITE port **local** **if**: it is found **in** the **local** part of its containing kernel structure; or **if** it is located **in** the foreign part of its containing kernel structure and the **local** and foreign Internet addresses are the same; or **if** it is located **in** the foreign part of its containing kernel structure and the foreign Internet address is `INADDR_LOOPBACK` (127.0.0.1). This rule may make `lsnf` ignore some foreign ports on machines with multiple interfaces when the foreign Internet address is on a different interface from the **local** one.

See the `lsnf` FAQ (The FAQ section gives its location.) **for** further discussion of portmapper registration reporting issues.

Portmapper registration reporting is supported only on

dialects that have RPC header files. (Some Linux distributions with Glibc 2.14 **do** not have them.) When portmapper registration reporting is supported, the -h or -? **help** output will show the +|-M option.

- n inhibits the conversion of network numbers to host names **for** network files. Inhibiting conversion may make lsof run faster. It is also useful when host name lookup is not working properly.
- N selects the listing of NFS files.
- o directs lsof to display file offset at all times. It causes the SIZE/OFF output column title to be changed to OFFSET. Note: on some UNIX dialects lsof can't obtain accurate or consistent file offset information from its kernel data sources, sometimes just **for** particular kinds of files (e.g., socket files.) Consult the lsof FAQ (The FAQ section gives its location.) **for** more information.

The -o and -s options are mutually exclusive; they can't both be specified. When neither is specified, lsof displays whatever value - size or offset - is appropriate and available **for** the **type** of the file.

- o o defines the number of decimal digits (o) to be printed after the ``0t'' **for** a file offset before the form is switched to ``0x...''. An o value of zero (unlimited) directs lsof to use the ``0t'' form **for** all offset output.

This option does NOT direct lsof to display offset at all **times**; specify -o (without a trailing number) to **do** that. -o o only specifies the number of digits after ``0t'' **in** either mixed size and offset or offset-only output. Thus, **for** example, to direct lsof to display offset at all **times** with a decimal digit count of 10, use:

```
-o -o 10
or
-o010
```

The default number of digits allowed after ``0t'' is normally 8, but may have been changed by the lsof builder. Consult the description of the -o o option **in** the output of the -h or -? option to determine the default that is **in** effect.

- 0 directs lsof to bypass the strategy it uses to avoid being blocked by some kernel operations - i.e., doing them **in** forked child processes. See the BLOCKS AND TIMEOUTS and

AVOIDING KERNEL BLOCKS sections **for** more information on kernel operations that may block lsof.

While use of this option will reduce lsof startup overhead, it may also cause lsof to hang when the kernel doesn't respond to a function. Use this option cautiously.

- p s excludes or selects the listing of files **for** the processes whose optional process IDentification (PID) numbers are **in** the comma-separated **set** s - e.g., ``123'' or ``123,^456''. (There should be no spaces **in** the set.)

PID numbers that begin with ``^' (negation) represent exclusions.

Multiple process ID numbers are joined **in** a single ORed **set** before participating **in** AND option selection. However, PID exclusions are applied without ORing or ANDing and take effect before other selection criteria are applied.

- P inhibits the conversion of port numbers to port names **for** network files. Inhibiting the conversion may make lsof run a little faster. It is also useful when port name lookup is not working properly.

+|-r [t[m<fmt>]]

puts lsof **in** repeat mode. There lsof lists open files as selected by other options, delays t seconds (default fifteen), **then** repeats the listing, delaying and listing repetitively **until** stopped by a condition defined by the prefix to the option.

If the prefix is a ``-', repeat mode is endless. Lsof must be terminated with an interrupt or quit signal.

If the prefix is ``+', repeat mode will end the first cycle no open files are listed - and of course when lsof is stopped with an interrupt or quit signal. When repeat mode ends because no files are listed, the process **exit** code will be zero **if** any open files were ever listed; one, **if** none were ever listed.

Lsof marks the end of each listing: **if** field output is **in** progress (the -F, option has been specified), the default marker is ``m'; otherwise the default marker is ``====='. The marker is followed by a NL character.

The optional "m<fmt>" argument specifies a format **for** the

marker line. The <fmt> characters following 'm' are interpreted as a format specification to the strftime(3) **function**, when both it and the localtime(3) **function** are available **in** the dialect's C library. Consult the strftime(3) documentation **for** what may appear **in** its format specification. Note that when field output is requested with the -F option, <fmt> cannot contain the NL format, ``%n''. Note also that when <fmt> contains spaces or other characters that affect the shell's interpretation of arguments, <fmt> must be quoted appropriately.

Repeat mode reduces lsof startup overhead, so it is more efficient to use this mode than to call lsof repetitively from a shell script, **for** example.

To use repeat mode most efficiently, accompany +|-r with specification of other lsof selection options, so the amount of kernel memory access lsof does will be kept to a minimum. Options that filter at the process level - e.g., -c, -g, -p, -u - are the most efficient selectors.

Repeat mode is useful when coupled with field output (see the -F, option description) and a supervising **awk** or Perl script, or a C program.

-R directs lsof to list the Parent Process IDentification number **in** the PPID column.

-s [p:s]

s alone directs lsof to display file size at all times. It causes the SIZE/OFF output column title to be changed to SIZE. If the file does not have a size, nothing is displayed.

The optional -s p:s form is available only **for** selected dialects, and only when the -h or -? **help** output lists it.

When the optional form is available, the s may be followed by a protocol name (p), either TCP or UDP, a colon (':') and a comma-separated protocol state name list, the option causes open TCP and UDP files to be excluded **if** their state name(s) are **in** the list (s) preceded by a '^'; or included **if** their name(s) are not preceded by a '^'.

Dialects that support this option may support only one protocol. When an unsupported protocol is specified, a message will be displayed indicating state names **for** the protocol are unavailable.

When an inclusion list is defined, only network files with state names **in** the list will be present **in** the lsof output. Thus, specifying one state name means that only network files with that lone state name will be listed.

Case is unimportant **in** the protocol or state names, but there may be no spaces and the colon (':') separating the protocol name (p) and the state name list (s) is required.

If only TCP and UDP files are to be listed, as controlled by the specified exclusions and inclusions, the -i option must be specified, too. If only a single protocol's files are to be listed, add its name as an argument to the -i option.

For example, to list only network files with TCP state LISTEN, use:

```
-iTCP -sTCP:LISTEN
```

Or, **for** example, to list network files with all UDP states except Idle, use:

```
-iUDP -sUDP:Idle
```

State names vary with UNIX dialects, so it's not possible to provide a **complete** list. Some common TCP state names are: CLOSED, IDLE, BOUND, LISTEN, ESTABLISHED, SYN_SENT, SYN_RCDV, ESTABLISHED, CLOSE_WAIT, FIN_WAIT1, CLOSING, LAST_ACK, FIN_WAIT_2, and TIME_WAIT. Two common UDP state names are Unbound and Idle.

See the lsof FAQ (The FAQ section gives its location.) **for** more information on how to use protocol state exclusion and inclusion, including examples.

The -o (without a following decimal digit count) and -s option (without a following protocol and state name list) are mutually exclusive; they can't both be specified. When neither is specified, lsof displays whatever value - size or offset - is appropriate and available **for** the **type** of file.

Since some types of files don't have **true** sizes - sockets, FIFOs, pipes, etc. - lsof displays **for** their sizes the content amounts **in** their associated kernel buffers, **if** possible.

-S [t] specifies an optional time-out seconds value **for** kernel functions - lstat(2), readlink(2), and stat(2) - that

might otherwise deadlock. The minimum **for** *t* is two; the default, fifteen; when no value is specified, the default is used.

See the BLOCKS AND TIMEOUTS section **for** more information.

-T [*t*] controls the reporting of some TCP/TPI information, also reported by `netstat(1)`, following the network addresses. In normal output the information appears **in** parentheses, each item except TCP or TPI state name identified by a keyword, followed by ``='`, separated from others by a single space:

```
<TCP or TPI state name>  
QR=<read queue length>  
QS=<send queue length>  
SO=<socket options and values>  
SS=<socket states>  
TF=<TCP flags and values>  
WR=<window read length>  
WW=<window write length>
```

Not all values are reported **for** all UNIX dialects. Items values (when available) are reported after the item name and `'='`.

When the field output mode is **in** effect (See OUTPUT FOR OTHER PROGRAMS.) each item appears as a field with a ``T'` leading character.

-T with no following key characters disables TCP/TPI information reporting.

-T with following characters selects the reporting of specific TCP/TPI information:

```
f    selects reporting of socket options,  
      states and values, and TCP flags and  
      values.  
q    selects queue length reporting.  
s    selects connection state reporting.  
w    selects window size reporting.
```

Not all selections are enabled **for** some UNIX dialects. State may be selected **for** all dialects and is reported by default. The `-h` or `-? help` output **for** the `-T` option will show what selections may be used with the UNIX dialect.

When `-T` is used to **select** information - i.e., it is followed by one or more selection characters - the

displaying of state is disabled by default, and it must be explicitly selected again **in** the characters following -T. (In effect, **then**, the default is equivalent to -Ts.) For example, **if** queue lengths and state are desired, use -Tqs.

Socket options, socket states, some socket values, TCP flags and one TCP value may be reported (when available **in** the UNIX dialect) **in** the form of the names that commonly appear after SO_, so_, SS_, TCP_ and TF_ **in** the dialect's header files - most often <sys/socket.h>, <sys/socketvar.h> and <netinet/tcp_var.h>. Consult those header files **for** the meaning of the flags, options, states and values.

``SO='' precedes socket options and values; ``SS='', socket states; and ``TF='', TCP flags and values.

If a flag or option has a value, the value will follow an '=' and the name -- e.g., ``SO=LINGER=5'', ``SO=QLIM=5'', ``TF=MSS=512''. The following seven values may be reported:

Name Reported	Description (Common Symbol)
KEEPALIVE	keep alive time (SO_KEEPAIVE)
LINGER	linger time (SO_LINGER)
MSS	maximum segment size (TCP_MAXSEG)
PQLEN	partial listen queue connections
QLEN	established listen queue connections
QLIM	established listen queue limit
RCVBUF	receive buffer length (SO_RCVBUF)
SNDBUF	send buffer length (SO_SNDBUF)

Details on what socket options and values, socket states, and TCP flags and values may be displayed **for** particular UNIX dialects may be found **in** the answer to the ``Why doesn't lsof report socket options, socket states, and TCP flags and values **for** my dialect?'' and ``Why doesn't lsof report the partial listen queue connection count **for** my dialect?'' questions **in** the lsof FAQ (The FAQ section gives its location.)

- t specifies that lsof should produce terse output with process identifiers only and no header - e.g., so that the output may be piped to **kill**(1). -t selects the -w option.
- u s selects the listing of files **for** the user whose **login** names or user ID numbers are **in** the comma-separated **set** s - e.g., ``abe'', or ``548,root''. (There should be no

spaces **in** the set.)

Multiple **login** names or user ID numbers are joined **in** a single ORed **set** before participating **in** AND option selection.

If a **login** name or user ID is preceded by a `^`, it becomes a negation - i.e., files of processes owned by the **login** name or user ID will never be listed. A negated **login** name or user ID selection is neither ANDed nor ORed with other selections; it is applied before all other selections and absolutely excludes the listing of the files of the process. For example, to direct `lsf` to exclude the listing of files belonging to root processes, specify `^-u^root` or `^-u^0`.

- U selects the listing of UNIX domain socket files.
- v selects the listing of `lsf` version information, including: revision number; when the `lsf` binary was constructed; who constructed the binary and where; the name of the compiler used to construct the `lsf` binary; the version number of the compiler when readily available; the compiler and loader flags used to construct the `lsf` binary; and system information, typically the output of `uname's -a` option.
- V directs `lsf` to indicate the items it was asked to list and failed to find - **command** names, file names, Internet addresses or files, **login** names, NFS files, PIDs, PGIDs, and UIDs.

When other options are ANDed to search options, or compile-time options restrict the listing of some files, `lsf` may not report that it failed to find a search item when an ANDed option or compile-time option prevents the listing of the open file containing the located search item.

For example, `lsf -V -iTCP@foobar -a -d 999` may not report a failure to locate open files at `^TCP@foobar` and may not list any, **if** none have a file descriptor number of 999. A similar situation arises when `HASSECURITY` and `HASNOSOCKSECURITY` are defined at compile time and they prevent the listing of open files.

- +|-w Enables (+) or disables (-) the suppression of warning messages.

The `lsf` builder may choose to have warning messages

disabled or enabled by default. The default warning message state is indicated **in** the output of the -h or -? option. Disabling warning messages when they are already disabled or enabling them when already enabled is acceptable.

The -t option selects the -w option.

-x [fl]

may accompany the +d and +D options to direct their processing to cross over symbolic links and/or file system mount points encountered when scanning the directory (+d) or directory tree (+D).

If -x is specified by itself without a following parameter, cross-over processing of both symbolic links and file system mount points is enabled. Note that when -x is specified without a parameter, the next argument must begin with '-' or '+'.

The optional 'f' parameter enables file system mount point cross-over processing; 'l', symbolic link cross-over processing.

The -x option may not be supplied without also supplying a +d or +D option.

-X This is a dialect-specific option.

AIX:

This IBM AIX RISC/System 6000 option requests the reporting of executed text file and shared library references.

WARNING: because this option uses the kernel readx() **function**, its use on a busy AIX system might cause an application process to hang so completely that it can neither be killed nor stopped. I have never seen this happen or had a report of its happening, but I think there is a remote possibility it could happen.

By default use of readx() is disabled. On AIX 5L and above lsof may need setuid-root permission to perform the actions this option requests.

The lsof builder may specify that the -X option be restricted to processes whose real UID is root. If that has been **done**, the -X option will not appear **in** the -h or -? **help** output unless the real UID of the lsof process is root. The default lsof distribution allows

any UID to specify -X, so by default it will appear **in** the **help** output.

When AIX readx() use is disabled, lsof may not be able to report information **for** all text and loader file references, but it may also avoid exacerbating an AIX kernel directory search kernel error, known as the Stale Segment ID bug.

The readx() **function**, used by lsof or any other program to access some sections of kernel virtual memory, can trigger the Stale Segment ID bug. It can cause the kernel's dir_search() **function** to believe erroneously that part of an **in**-memory copy of a file system directory has been zeroed. Another application process, distinct from lsof, asking the kernel to search the directory - e.g., by using open(2) - can cause dir_search() to loop forever, thus hanging the application process.

Consult the lsof FAQ (The FAQ section gives its location.) and the 00README file of the lsof distribution **for** a more **complete** description of the Stale Segment ID bug, its APAR, and methods **for** defining readx() use when compiling lsof.

Linux:

This Linux option requests that lsof skip the reporting of information on all open TCP, UDP and UDPLITE IPv4 and IPv6 files.

This Linux option is most useful when the system has an extremely large number of open TCP, UDP and UDPLITE files, the processing of whose information **in** the /proc/net/tcp* and /proc/net/udp* files would take lsof a long time, and whose reporting is not of interest.

Use this option with care and only when you are sure that the information you want lsof to display isn't associated with open TCP, UDP or UDPLITE socket files.

Solaris 10 and above:

This Solaris 10 and above option requests the reporting of cached paths **for** files that have been deleted - i.e., removed with rm(1) or unlink(2).

The cached path is followed by the string `` (deleted)'' to indicate that the path by which the file was opened has been deleted.

Because intervening changes made to the path - i.e., renames with mv(1) or rename(2) - are not recorded **in** the cached path, what lsof reports is only the path by which the file was opened, not its possibly different final path.

-z [z] specifies how Solaris 10 and higher zone information is to be handled.

Without a following argument - e.g., NO z - the option specifies that zone names are to be listed **in** the ZONE output column.

The -z option may be followed by a zone name, z. That causes lsof to list only open files **for** processes **in** that zone. Multiple -z z option and argument pairs may be specified to form a list of named zones. Any open file of any process **in** any of the zones will be listed, subject to other conditions specified by other options and arguments.

-Z [Z] specifies how SELinux security contexts are to be handled. It and 'Z' field output character support are inhibited when SELinux is disabled **in** the running Linux kernel. See OUTPUT FOR OTHER PROGRAMS **for** more information on the 'Z' field output character.

Without a following argument - e.g., NO Z - the option specifies that security contexts are to be listed **in** the SECURITY-CONTEXT output column.

The -Z option may be followed by a wildcard security context name, Z. That causes lsof to list only open files **for** processes **in** that security context. Multiple -Z Z option and argument pairs may be specified to form a list of security contexts. Any open file of any process **in** any of the security contexts will be listed, subject to other conditions specified by other options and arguments. Note that Z can be A:B:C or *:B:C or A:B:* or *:*:C to match against the A:B:C context.

-- The double minus sign option is a marker that signals the end of the keyed options. It may be used, **for** example, when the first file name begins with a minus sign. It may also be used when the absence of a value **for** the last keyed option must be signified by the presence of a minus sign **in** the following option and before the start of the file names.

names These are path names of specific files to list.

Symbolic links are resolved before use. The first name may be separated from the preceding options with the `--''` option.

If a name is the mounted-on directory of a file system or the device of the file system, `lsf` will list all the files open on the file system. To be considered a file system, the name must match a mounted-on directory name **in** `mount(8)` output, or match the name of a block device associated with a mounted-on directory name. The `+|-f` option may be used to force `lsf` to consider a name a file system identifier (`+f`) or a simple file (`-f`).

If name is a path to a directory that is not the mounted-on directory name of a file system, it is treated just as a regular file is treated - i.e., its listing is restricted to processes that have it open as a file or as a process-specific directory, such as the root or current working directory. To request that `lsf` look **for** open files inside a directory name, use the `+d s` and `+D D` options.

If a name is the base name of a family of multiplexed files - e.g, AIX's `/dev/pt[cs]` - `lsf` will list all the associated multiplexed files on the device that are open - e.g., `/dev/pt[cs]/1`, `/dev/pt[cs]/2`, etc.

If a name is a UNIX domain socket name, `lsf` will usually search **for** it by the characters of the name alone - exactly as it is specified and is recorded **in** the kernel socket structure. (See the next paragraph **for** an exception to that rule **for** Linux.) Specifying a relative path - e.g., `./file` - **in** place of the file's absolute path - e.g., `/tmp/file` - won't work because `lsf` must match the characters you specify with what it finds **in** the kernel UNIX domain socket structures.

If a name is a Linux UNIX domain socket name, **in** one **case** `lsf` is able to search **for** it by its device and inode number, allowing name to be a relative path. The **case** requires that the absolute path -- i.e., one beginning with a slash (`'/'`) be used by the process that created the socket, and hence be stored **in** the `/proc/net/unix` file; and it requires that `lsf` be able to obtain the device and node numbers of both the absolute path **in** `/proc/net/unix` and name via successful `stat(2)` system calls. When those conditions are met, `lsf` will be able to search **for** the UNIX domain socket when some path to it is specified **in** name. Thus, **for** example, **if** the path is `/dev/log`, and an `lsf` search is

initiated when the working directory is /dev, **then** name could be ./log.

If a name is none of the above, lsof will list any open files whose device and inode match that of the specified path name.

If you have also specified the -b option, the only names you may safely specify are file systems **for** which your mount table supplies alternate device numbers. See the AVOIDING KERNEL BLOCKS and ALTERNATE DEVICE NUMBERS sections **for** more information.

Multiple file names are joined **in** a single ORed **set** before participating **in** AND option selection.

AFS

Lsof supports the recognition of AFS files **for** these dialects (and AFS versions):

- AIX 4.1.4 (AFS 3.4a)
- HP-UX 9.0.5 (AFS 3.4a)
- Linux 1.2.13 (AFS 3.3)
- Solaris 2.[56] (AFS 3.4a)

It may recognize AFS files on other versions of these dialects, but has not been tested there. Depending on how AFS is implemented, lsof may recognize AFS files **in** other dialects, or may have difficulties recognizing AFS files **in** the supported dialects.

Lsof may have trouble identifying all aspects of AFS files **in** supported dialects when AFS kernel support is implemented via dynamic modules whose addresses **do** not appear **in** the kernel's variable name list. In that **case**, lsof may have to guess at the identity of AFS files, and might not be able to obtain volume information from the kernel that is needed **for** calculating AFS volume node numbers. When lsof can't compute volume node numbers, it reports blank **in** the NODE column.

The -A A option is available **in** some dialect implementations of lsof **for** specifying the name list file where dynamic module kernel addresses may be found. When this option is available, it will be listed **in** the lsof **help** output, presented **in** response to the -h or -?

See the lsof FAQ (The FAQ section gives its location.) **for** more information about dynamic modules, their symbols, and how they affect lsof options.

Because AFS path lookups don't seem to participate **in** the

kernel's name cache operations, lsof can't identify path name components **for** AFS files.

SECURITY

Lsof has three features that may cause security concerns. First, its default compilation mode allows anyone to list all open files with it. Second, by default it creates a user-readable and user-writable device cache file **in** the home directory of the real user ID that executes lsof. (The list-all-open-files and device cache features may be disabled when lsof is compiled.) Third, its **-k** and **-m** options name alternate kernel name list or memory files.

Restricting the listing of all open files is controlled by the compile-time HASSECURITY and HASNOSOCKSECURITY options. When HASSECURITY is defined, lsof will allow only the root user to list all open files. The non-root user may list only open files of processes with the same user IDentification number as the real user ID number of the lsof process (the one that its user logged on with).

However, **if** HASSECURITY and HASNOSOCKSECURITY are both defined, anyone may list open socket files, provided they are selected with the **-i** option.

When HASSECURITY is not defined, anyone may list all open files.

Help output, presented **in** response to the **-h** or **-?** option, gives the status of the HASSECURITY and HASNOSOCKSECURITY definitions.

See the Security section of the 00README file of the lsof distribution **for** information on building lsof with the HASSECURITY and HASNOSOCKSECURITY options enabled.

Creation and use of a user-readable and user-writable device cache file is controlled by the compile-time HASDCACHE option. See the DEVICE CACHE FILE section and the sections that follow it **for** details on how its path is formed. For security considerations it is important to note that **in** the default lsof distribution, **if** the real user ID under which lsof is executed is root, the device cache file will be written **in** root's home directory - e.g., **/** or **/root**. When HASDCACHE is not defined, lsof does not write or attempt to **read** a device cache file.

When HASDCACHE is defined, the lsof **help** output, presented **in** response to the **-h**, **-D?**, or **-?** options, will provide device cache file handling information. When HASDCACHE is not defined, the **-h** or **-?** output will have no **-D** option description.

Before you decide to disable the device cache file feature - enabling it improves the performance of lsof by reducing the

startup overhead of examining all the nodes **in** /dev (or /devices)
- **read** the discussion of it **in** the 00DCACHE file of the lsof
distribution and the lsof FAQ (The FAQ section gives its
location.)

WHEN IN DOUBT, YOU CAN TEMPORARILY DISABLE THE USE OF THE DEVICE
CACHE FILE WITH THE -Di OPTION.

When lsof user declares alternate kernel name list or memory
files with the -k and -m options, lsof checks the user's
authority to **read** them with access(2). This is intended to
prevent whatever special power lsof's modes might confer on it
from letting it **read** files not normally accessible via the
authority of the real user ID.

OUTPUT

This section describes the information lsof lists **for** each open
file. See the OUTPUT FOR OTHER PROGRAMS section **for** additional
information on output that can be processed by another program.

Lsof only outputs printable (declared so by isprint(3)) 8 bit
characters. Non-printable characters are printed **in** one of three
forms: the C `\\[bfrnt]'` form; the control character `^'` form
(e.g., `^^@'`); or hexadecimal leading `\\x'` form (e.g.,
`\\xab'`). Space is non-printable **in** the COMMAND column
(`\\x20'`) and printable elsewhere.

For some dialects - **if** HASSETLOCALE is defined **in** the dialect's
machine.h header file - lsof will print the extended 8 bit
characters of a language locale. The lsof process must be
supplied a language locale environment variable (e.g., LANG)
whose value represents a known language locale **in** which the
extended characters are considered printable by isprint(3).
Otherwise lsof considers the extended characters non-printable
and prints them according to its rules **for** non-printable
characters, stated above. Consult your dialect's setlocale(3)
man page **for** the names of other environment variables that may be
used **in** place of LANG - e.g., LC_ALL, LC_CTYPE, etc.

Lsof's language locale support **for** a dialect also covers wide
characters - e.g., UTF-8 - when HASSETLOCALE and HASWIDECHAR are
defined **in** the dialect's machine.h header file, and when a
suitable language locale has been defined **in** the appropriate
environment variable **for** the lsof process. Wide characters are
printable under those conditions **if** iswprint(3) reports them to
be. If HASSETLOCALE, HASWIDECHAR and a suitable language locale
aren't defined, or **if** iswprint(3) reports wide characters that
aren't printable, lsof considers the wide characters
non-printable and prints each of their 8 bits according to its
rules **for** non-printable characters, stated above.

Consult the answers to the "Language_locale_support" questions **in** the lsof FAQ (The FAQ section gives its location.) **for** more information.

Lsof dynamically sizes the output columns each time it runs, guaranteeing that each column is a minimum size. It also guarantees that each column is separated from its predecessor by at least one space.

COMMAND

contains the first nine characters of the name of the UNIX **command** associated with the process. If a non-zero w value is specified to the +c w option, the column contains the first w characters of the name of the UNIX **command** associated with the process up to the limit of characters supplied to lsof by the UNIX dialect. (See the description of the +c w **command** or the lsof FAQ **for** more information. The FAQ section gives its location.)

If w is less than the length of the column title, ``COMMAND'', it will be raised to that length.

If a zero w value is specified to the +c w option, the column contains all the characters of the name of the UNIX **command** associated with the process.

All **command** name characters maintained by the kernel **in** its structures are displayed **in** field output when the **command** name descriptor ('\c') is specified. See the OUTPUT FOR OTHER COMMANDS section **for** information on selecting field output and the associated **command** name descriptor.

PID is the Process IDentification number of the process.

TID is the task (thread) IDentification number, **if** task (thread) reporting is supported by the dialect and a task (thread) is being listed. (If **help** output - i.e., the output of the -h or -? options - shows this option, **then** task (thread) reporting is supported by the dialect.)

A blank TID column **in** Linux indicates a process - i.e., a non-task.

TASKCMD

is the task **command** name. Generally this will be the same as the process named **in** the COMMAND column, but some task implementations (e.g., Linux) permit a task to change its **command** name.

The TASKCMD column width is subject to the same size limitation as the COMMAND column.

ZONE is the Solaris 10 and higher zone name. This column must be selected with the -z option.

SECURITY-CONTEXT

is the SELinux security context. This column must be selected with the -Z option. Note that the -Z option is inhibited when SELinux is disabled in the running Linux kernel.

PPID is the Parent Process IDentification number of the process. It is only displayed when the -R option has been specified.

PGID is the process group IDentification number associated with the process. It is only displayed when the -g option has been specified.

USER is the user ID number or **login** name of the user to whom the process belongs, usually the same as reported by **ps(1)**. However, on Linux USER is the user ID number or **login** that owns the directory in /proc where lsof finds information about the process. Usually that is the same value reported by **ps(1)**, but may differ when the process has changed its effective user ID. (See the -l option description for information on when a user ID number or **login** name is displayed.)

FD is the File Descriptor number of the file or:

- cwd current working directory;
- Lnn library references (AIX);
- err FD information error (see NAME column);
- jld jail directory (FreeBSD);
- ltx shared library text (code and data);
- Mxx hex memory-mapped **type** number xx.
- m86 DOS Merge mapped file;
- mem memory-mapped file;
- mmap memory-mapped device;
- pd parent directory;
- rtd root directory;
- tr kernel trace file (OpenBSD);
- txt program text (code and data);
- v86 VP/ix mapped file;

FD is followed by one of these characters, describing the mode under which the file is open:

```

r for read access;
w for write access;
u for read and write access;
space if mode unknown and no lock
    character follows;
`-' if mode unknown and lock
    character follows.

```

The mode character is followed by one of these lock characters, describing the **type** of lock applied to the file:

```

N for a Solaris NFS lock of unknown type;
r for read lock on part of the file;
R for a read lock on the entire file;
w for a write lock on part of the file;
W for a write lock on the entire file;
u for a read and write lock of any length;
U for a lock of unknown type;
x for an SCO OpenServer Xenix lock on part      of
the file;
X for an SCO OpenServer Xenix lock on the entire
file;
space if there is no lock.

```

See the LOCKS section **for** more information on the lock information character.

The FD column contents constitutes a single field **for** parsing **in** post-processing scripts.

TYPE is the **type** of the node associated with the file - e.g., GDIR, GREG, VDIR, VREG, etc.

or ``IPv4'' **for** an IPv4 socket;

or ``IPv6'' **for** an open IPv6 network file - even **if** its address is IPv4, mapped **in** an IPv6 address;

or ``ax25'' **for** a Linux AX.25 socket;

or ``inet'' **for** an Internet domain socket;

or ``lla'' **for** a HP-UX link level access file;

or ``rte'' **for** an AF_ROUTE socket;

or ``sock'' **for** a socket of unknown domain;

or ``unix'' **for** a UNIX domain socket;

or ``x.25'' **for** an HP-UX x.25 socket;

or ``BLK'' **for** a block special file;

or ``CHR'' **for** a character special file;

or ``DEL'' **for** a Linux map file that has been deleted;

or ``DIR'' **for** a directory;

or ``D00R'' **for** a VD00R file;

or ``FIFO'' **for** a FIFO special file;

or ``KQUEUE'' **for** a BSD style kernel event queue file;

or ``LINK'' **for** a symbolic link file;

or ``MPB'' **for** a multiplexed block file;

or ``MPC'' **for** a multiplexed character file;

or ``NOFD'' **for** a Linux /proc/<PID>/fd directory that can't be opened -- the directory path appears **in** the NAME column, followed by an error message;

or ``PAS'' **for** a /proc/as file;

or ``PAXV'' **for** a /proc/auxv file;

or ``PCRE'' **for** a /proc/cred file;

or ``PCTL'' **for** a /proc control file;

or ``PCUR'' **for** the current /proc process;

or ``PCWD'' **for** a /proc current working directory;

or ``PDIR'' **for** a /proc directory;

or ``PETY'' **for** a /proc executable **type** (etype);

or ``PFD'' **for** a /proc file descriptor;

or ``PFDR'' **for** a /proc file descriptor directory;

or ``PFIL'' **for** an executable /proc file;

or ``PFPR'' **for** a /proc FP register **set**;

or ``PGD'' **for** a /proc/pagedata file;
or ``PGID'' **for** a /proc group notifier file;
or ``PIPE'' **for** pipes;
or ``PLC'' **for** a /proc/lwpctl file;
or ``PLDR'' **for** a /proc/lpw directory;
or ``PLDT'' **for** a /proc/ldt file;
or ``PLPI'' **for** a /proc/lpsinfo file;
or ``PLST'' **for** a /proc/lstatus file;
or ``PLU'' **for** a /proc/lusage file;
or ``PLWG'' **for** a /proc/gwindows file;
or ``PLWI'' **for** a /proc/lwpsinfo file;
or ``PLWS'' **for** a /proc/lwpstatus file;
or ``PLWU'' **for** a /proc/lwpusage file;
or ``PLWX'' **for** a /proc/xregs file;
or ``PMAP'' **for** a /proc map file (map);
or ``PMEM'' **for** a /proc memory image file;
or ``PNTF'' **for** a /proc process notifier file;
or ``POBJ'' **for** a /proc/object file;
or ``PODR'' **for** a /proc/object directory;
or ``POLP'' **for** an old format /proc light weight process
file;
or ``POPF'' **for** an old format /proc PID file;
or ``POPG'' **for** an old format /proc page data file;
or ``PORT'' **for** a SYSV named pipe;
or ``PREG'' **for** a /proc register file;

or ``PRMP'' **for** a /proc/rmap file;
 or ``PRTD'' **for** a /proc root directory;
 or ``PSGA'' **for** a /proc/sigact file;
 or ``PSIN'' **for** a /proc/psinfo file;
 or ``PSTA'' **for** a /proc status file;
 or ``PSXSEM'' **for** a POSIX semaphore file;
 or ``PSXSHM'' **for** a POSIX shared memory file;
 or ``PTS'' **for** a /dev/pts file;
 or ``PUSG'' **for** a /proc/usage file;
 or ``PW'' **for** a /proc/watch file;
 or ``PXMP'' **for** a /proc/xmap file;
 or ``REG'' **for** a regular file;
 or ``SMT'' **for** a shared memory transport file;
 or ``STSO'' **for** a stream socket;
 or ``UNNM'' **for** an unnamed **type** file;
 or ``XNAM'' **for** an OpenServer Xenix special file of
 unknown **type**;
 or ``XSEM'' **for** an OpenServer Xenix semaphore file;
 or ``XSD'' **for** an OpenServer Xenix shared data file;

 or the four **type** number octets **if** the corresponding name
 isn't known.

FILE-ADDR

contains the kernel file structure address when f has been specified to +f;

FCT

contains the file reference count from the kernel file structure when c has been specified to +f;

FILE-FLAG

when g or G has been specified to +f, this field contains the contents of the f_flag[s] member of the kernel file

structure and the kernel's per-process open file flags (**if** available); **`G'** causes them to be displayed **in** hexadecimal; **`g'**, as short-hand names; two lists may be displayed with entries separated by commas, the lists separated by a semicolon (**`;'**); the first list may contain short-hand names **for** **f_flag[s]** values from the following table:

AIO	asynchronous I/O (e.g., FAIO)
AP	append
ASYN	asynchronous I/O (e.g., FASYN)
BAS	block, test , and set in use
BKIU	block if in use
BL	use block offsets
BSK	block seek
CA	copy avoid
CIO	concurrent I/O
CLON	clone
CLRD	CL read
CR	create
DF	defer
DFI	defer IND
DFLU	data flush
DIR	direct
DLY	delay
DOCL	do clone
DSYN	data-only integrity
DTY	must be a directory
EVO	event only
EX	open for exec
EXCL	exclusive open
FSYN	synchronous writes
GCDF	defer during unp_gc() (AIX)
GCMK	mark during unp_gc() (AIX)
GTTY	accessed via /dev/tty
HUP	HUP in progress
KERN	kernel
KIOC	kernel-issued ioctl
LCK	has lock
LG	large file
MBLK	stream message block
MK	mark
MNT	mount
MSYN	multiplex synchronization
NATM	don't update atime
NB	non-blocking I/O
NBDR	no BDRM check
NBIO	SYSV non-blocking I/O
NBF	n-buffering in effect
NC	no cache

ND	no delay
NDSY	no data synchronization
NET	network
NFLK	don't follow links
NMFS	NM file system
NOTO	disable background stop
NSH	no share
NTTY	no controlling TTY
OLRM	OLR mirror
PAIO	POSIX asynchronous I/O
PP	POSIX pipe
R	read
RC	file and record locking cache
REV	revoked
RSH	shared read
RSYN	read synchronization
RW	read and write access
SL	shared lock
SNAP	cooked snapshot
SOCK	socket
SQSH	Sequent shared set on open
SQSV	Sequent SVM set on open
SQR	Sequent set repair on open
SQS1	Sequent full shared open
SQS2	Sequent partial shared open
STPI	stop I/O
SWR	synchronous read
SYN	file integrity while writing
TCPM	avoid TCP collision
TR	truncate
W	write
WKUP	parallel I/O synchronization
WTG	parallel I/O synchronization
VH	vhangup pending
VTXT	virtual text
XL	exclusive lock

this list of names was derived from F* *#define's* in dialect header files <fcntl.h>, <linux/fs.h>, <sys/fcntl.c>, <sys/fcntlcom.h>, and <sys/file.h>; see the lsof.h header file **for** a list showing the correspondence between the above short-hand names and the header file definitions;

the second list (after the semicolon) may contain short-hand names **for** kernel per-process open file flags from this table:

ALLC	allocated
BR	the file has been read

BHUP	activity stopped by SIGHUP
BW	the file has been written
CLSG	closing
CX	close-on- exec (see fcntl(F_SETFD))
LCK	lock was applied
MP	memory-mapped
OPIP	open pending - in progress
RSVW	reserved wait
SHMT	UF_FSHMAT set (AIX)
USE	in use (multi-threaded)

NODE-ID

(or INODE-ADDR **for** some dialects) contains a unique identifier **for** the file node (usually the kernel vnode or inode address, but also occasionally a concatenation of device and node number) when n has been specified to +f;

DEVICE contains the device numbers, separated by commas, **for** a character special, block special, regular, directory or NFS file;

or ``memory'' **for** a memory file system node under Tru64 UNIX;

or the address of the private data area of a Solaris socket stream;

or a kernel reference address that identifies the file (The kernel reference address may be used **for** FIFO's, **for** example.);

or the base address or device name of a Linux AX.25 socket device.

Usually only the lower thirty two bits of Tru64 UNIX kernel addresses are displayed.

SIZE, SIZE/OFF, or OFFSET

is the size of the file or the file offset **in** bytes. A value is displayed **in** this column only **if** it is available. Lsof displays whatever value - size or offset - is appropriate **for** the **type** of the file and the version of lsof.

On some UNIX dialects lsof can't obtain accurate or consistent file offset information from its kernel data sources, sometimes just **for** particular kinds of files (e.g., socket files.) In other cases, files don't have **true** sizes - e.g., sockets, FIFOs, pipes - so lsof displays **for** their sizes the content amounts it finds **in**

their kernel buffer descriptors (e.g., socket buffer size counts or TCP/IP window sizes.) Consult the lsof FAQ (The FAQ section gives its location.) **for** more information.

The file size is displayed **in** decimal; the offset is normally displayed **in** decimal with a leading ``0t'' **if** it contains 8 digits or less; **in** hexadecimal with a leading ``0x'' **if** it is longer than 8 digits. (Consult the -o o option description **for** information on when 8 might default to some other value.)

Thus the leading ``0t'' and ``0x'' identify an offset when the column may contain both a size and an offset (i.e., its title is SIZE/OFF).

If the -o option is specified, lsof always displays the file offset (or nothing **if** no offset is available) and labels the column OFFSET. The offset always begins with ``0t'' or ``0x'' as described above.

The lsof user can control the switch from ``0t'' to ``0x'' with the -o o option. Consult its description **for** more information.

If the -s option is specified, lsof always displays the file size (or nothing **if** no size is available) and labels the column SIZE. The -o and -s options are mutually exclusive; they can't both be specified.

For files that don't have a fixed size - e.g., don't reside on a disk device - lsof will display appropriate information about the current size or position of the file **if** it is available **in** the kernel structures that define the file.

NLINK contains the file link count when +L has been specified;

NODE is the node number of a **local** file;

or the inode number of an NFS file **in** the server host;

or the Internet protocol **type** - e.g., ``TCP'';

or ``STR'' **for** a stream;

or ``CCITT'' **for** an HP-UX x.25 socket;

or the IRQ or inode number of a Linux AX.25 socket device.

NAME is the name of the mount point and file system on which

the file resides;

or the name of a file specified **in** the names option (after any symbolic links have been resolved);

or the name of a character special or block special device;

or the **local** and remote Internet addresses of a network file; the **local** host name or IP number is followed by a colon (':'), the port, ``->', and the two-part remote address; IP addresses may be reported as numbers or names, depending on the +|-M, -n, and -P options; colon-separated IPv6 numbers are enclosed **in** square brackets; IPv4 INADDR_ANY and IPv6 IN6_IS_ADDR_UNSPECIFIED addresses, and zero port numbers are represented by an asterisk (*); a UDP destination address may be followed by the amount of time elapsed since the last packet was sent to the destination; TCP, UDP and UDPLITE remote addresses may be followed by TCP/TPI information **in** parentheses - state (e.g., ``(ESTABLISHED)'`, ``(Unbound)'`), queue sizes, and window sizes (not all dialects) - **in** a fashion similar to what netstat(1) reports; see the -T option description or the description of the TCP/TPI field **in** OUTPUT FOR OTHER PROGRAMS **for** more information on state, queue size, and window size;

or the address or name of a UNIX domain socket, possibly including a stream clone device name, a file system object's path name, **local** and foreign kernel addresses, socket pair information, and a bound vnode address;

or the **local** and remote mount point names of an NFS file;

or ``STR'', followed by the stream name;

or a stream character device name, followed by ``->'`' and the stream name or a list of stream module names, separated by ``->'`';

or ``STR:'`' followed by the SCO OpenServer stream device and module names, separated by ``->'`';

or system directory name, `` -- ''', and as many components of the path name as lsof can find **in** the kernel's name cache **for** selected dialects (See the KERNEL NAME CACHE section **for** more information.);

or ``PIPE->'`, followed by a Solaris kernel pipe destination address;

or ``COMMON:'' , followed by the vnode device information structure's device name, **for** a Solaris common vnode;

or the address family, followed by a slash ('/'), followed by fourteen comma-separated bytes of a non-Internet raw socket address;

or the HP-UX x.25 **local** address, followed by the virtual connection number (**if** any), followed by the remote address (**if** any);

or ``(dead)'' **for** disassociated Tru64 UNIX files - typically terminal files that have been flagged with the TIOCNOTTY ioctl and closed by daemons;

or ``rd=<offset>'' and ``wr=<offset>'' **for** the values of the **read** and write offsets of a FIFO;

or ``clone n:/dev/event'' **for** SCO OpenServer file clones of the /dev/event device, where n is the minor device number of the file;

or ``(socketpair: n)'' **for** a Solaris 2.6, 8, 9 or 10 UNIX domain socket, created by the socketpair(3N) network **function**;

or ``no PCB'' **for** socket files that **do** not have a protocol block associated with them, optionally followed by `` , CANTSENDMORE'' **if** sending on the socket has been disabled, or `` , CANTRCVMORE'' **if** receiving on the socket has been disabled (e.g., by the shutdown(2) **function**);

or the **local** and remote addresses of a Linux IPX socket file **in** the form <net>:[<node>:]<port>, followed **in** parentheses by the transmit and receive queue sizes, and the connection state;

or ``dgram'' or ``stream'' **for** the **type** UnixWare 7.1.1 and above **in**-kernel UNIX domain sockets, followed by a colon (':') and the **local** path name when available, followed by ``->'' and the remote path name or kernel socket address **in** hexadecimal when available;

or the association value, association index, endpoint value, **local** address, **local** port, remote address and remote port **for** Linux SCTP sockets;

or ``protocol: '' followed by the Linux socket's protocol attribute.

For dialects that support a ``namefs'' file system, allowing one file to be attached to another with `fattach(3C)`, `lsf` will add ``(FA:<address1><direction><address2>)' to the NAME column. <address1> and <address2> are hexadecimal vnode addresses. <direction> will be ``<-'' if <address2> has been `fattach`'ed to this vnode whose address is <address1>; and ``->'' if <address1>, the vnode address of this vnode, has been `fattach`'ed to <address2>. <address1> may be omitted if it already appears in the DEVICE column.

`lsf` may add two parenthetical notes to the NAME column for open Solaris 10 files: ``(?)'' if `lsf` considers the path name of questionable accuracy; and ``(deleted)'' if the `-X` option has been specified and `lsf` detects the open file's path name has been deleted. Consult the `lsf` FAQ (The FAQ section gives its location.) for more information on these NAME column additions.

LOCKS

`lsf` can't adequately report the wide variety of UNIX dialect file locks in a single character. What it reports in a single character is a compromise between the information it finds in the kernel and the limitations of the reporting format.

Moreover, when a process holds several byte level locks on a file, `lsf` only reports the status of the first lock it encounters. If it is a byte level lock, then the lock character will be reported in lower case - i.e., ``r'`, ``w'`, or ``x'` - rather than the upper case equivalent reported for a full file lock.

Generally `lsf` can only report on locks held by local processes on local files. When a local process sets a lock on a remotely mounted (e.g., NFS) file, the remote server host usually records the lock state. One exception is Solaris - at some patch levels of 2.3, and in all versions above 2.4, the Solaris kernel records information on remote locks in local structures.

`lsf` has trouble reporting locks for some UNIX dialects. Consult the BUGS section of this manual page or the `lsf` FAQ (The FAQ section gives its location.) for more information.

OUTPUT FOR OTHER PROGRAMS

When the `-F` option is specified, `lsf` produces output that is suitable for processing by another program - e.g., an `awk` or Perl script, or a C program.

Each unit of information is output in a field that is identified with a leading character and terminated by a NL (012) (or a NUL (000) if the 0 (zero) field identifier character is specified.) The data of the field follows immediately after the field identification character and extends to the field terminator.

It is possible to think of field output as process and file sets. A process **set** begins with a field whose identifier is ``p'` (**f**or process IDentifier (PID)). It extends to the beginning of the next PID field or the beginning of the first file **set** of the process, whichever comes first. Included **in** the process **set** are fields that identify the **command**, the process group IDentification (PGID) number, the task (thread) ID (TID), and the user ID (UID) number or **login** name.

A file **set** begins with a field whose identifier is ``f'` (**f**or file descriptor). It is followed by lines that describe the file's access mode, lock state, **type**, device, size, offset, inode, protocol, name and stream module names. It extends to the beginning of the next file or process **set**, whichever comes first.

When the NUL (000) field terminator has been selected with the 0 (zero) field identifier character, `lsOf` ends each process and file **set** with a NL (012) character.

`lsOf` always produces one field, the PID (``p'`) field. All other fields may be declared optionally **in** the field identifier character list that follows the `-F` option. When a field selection character identifies an item `lsOf` does not normally list - e.g., PPID, selected with `-R` - specification of the field character - e.g., ```-FR''` - also selects the listing of the item.

It is entirely possible to **select** a **set** of fields that cannot easily be parsed - e.g., **if** the field descriptor field is not selected, it may be difficult to identify file sets. To **help** you avoid this difficulty, `lsOf` supports the `-F` option; it selects the output of all fields with NL terminators (the `-F0` option pair selects the output of all fields with NUL terminators). For compatibility reasons neither `-F` nor `-F0` **select** the raw device field.

These are the fields that `lsOf` will produce. The single character listed first is the field identifier.

a	file access mode
c	process command name (all characters from proc or user structure)
C	file structure share count
d	file's device character code
D	file's major/minor device number (0x<hexadecimal>)
f	file descriptor (always selected)
F	file structure address (0x<hexadecimal>)
G	file flaGs (0x<hexadecimal>; names if + fg follows)
g	process group ID
i	file's inode number
K	task ID

```

k    link count
l    file's lock status
L    process login name
m    marker between repeated output
M    the task comMand name
n    file name, comment, Internet address
N    node identifier (0x<hexadecimal>)
o    file's offset (decimal)
p    process ID (always selected)
P    protocol name
r    raw device number (0x<hexadecimal>)
R    parent process ID
s    file's size (decimal)
S    file's stream identification
t    file's type
T    TCP/TPI information, identified by prefixes (the
    '=' is part of the prefix):
    QR=<read queue size>
    QS=<send queue size>
    SO=<socket options and values> (not all dialects)
    SS=<socket states> (not all dialects)
    ST=<connection state>
    TF=<TCP flags and values> (not all dialects)
    WR=<window read size> (not all dialects)
    WW=<window write size> (not all dialects)
    (TCP/TPI information isn't reported for all supported
    UNIX dialects. The -h or -? help output for the
    -T option will show what TCP/TPI reporting can be
    requested.)
u    process user ID
z    Solaris 10 and higher zone name
Z    SELinux security context (inhibited when SELinux is disabled)
0    use NUL field terminator character in place of NL
1-9  dialect-specific field identifiers (The output
    of -F? identifies the information to be found
    in dialect-specific fields.)

```

You can get on-line **help** information on these characters and their descriptions by specifying the -F? option pair. (Escape the '?' character as your shell requires.) Additional information on field content can be found **in** the OUTPUT section.

As an example, ``-F pcfn'' will **select** the process ID ('p'), **command** name ('c'), file descriptor ('f') and file name ('n') fields with an NL field terminator character; ``-F pcfn0'' selects the same output with a NUL (000) field terminator character.

Lsof doesn't produce all fields **for** every process or file **set**, only those that are available. Some fields are mutually

exclusive: file device characters and file major/minor device numbers; file inode number and protocol name; file name and stream identification; file size and offset. One or the other member of these mutually exclusive sets will appear **in** field output, but not both.

Normally lsof ends each field with a NL (012) character. The 0 (zero) field identifier character may be specified to change the field terminator character to a NUL (000). A NUL terminator may be easier to process with xargs(1), **for** example, or with programs whose quoting mechanisms may not easily cope with the range of characters **in** the field output. When the NUL field terminator is **in** use, lsof ends each process and file **set** with a NL (012).

Three aids to producing programs that can process lsof field output are included **in** the lsof distribution. The first is a C header file, lsof_fields.h, that contains symbols **for** the field identification characters, indexes **for** storing them **in** a table, and explanation strings that may be compiled into programs. Lsof uses this header file.

The second aid is a **set** of sample scripts that process field output, written **in** awk, Perl 4, and Perl 5. They're located **in** the scripts subdirectory of the lsof distribution.

The third aid is the C library used **for** the lsof **test** suite. The **test** suite is written **in** C and uses field output to validate the correct operation of lsof. The library can be found **in** the tests/LTlib.c file of the lsof distribution. The library uses the first aid, the lsof_fields.h header file.

BLOCKS AND TIMEOUTS

Lsof can be blocked by some kernel functions that it uses - lstat(2), readlink(2), and stat(2). These functions are stalled **in** the kernel, **for** example, when the hosts where mounted NFS file systems reside become inaccessible.

Lsof attempts to **break** these blocks with timers and child processes, but the techniques are not wholly reliable. When lsof does manage to **break** a block, it will report the **break** with an error message. The messages may be suppressed with the -t and -w options.

The default timeout value may be displayed with the -h or -? option, and it may be changed with the -S [t] option. The minimum **for** t is two seconds, but you should avoid small values, since slow system responsiveness can cause short timeouts to expire unexpectedly and perhaps stop lsof before it can produce any output.

When lsof has to **break** a block during its access of mounted file

system information, it normally continues, although with less information available to display about open files.

Lsof can also be directed to avoid the protection of timers and child processes when using the kernel functions that might block by specifying the `-O` option. While this will allow lsof to start up with less overhead, it exposes lsof completely to the kernel situations that might block it. Use this option cautiously.

AVOIDING KERNEL BLOCKS

You can use the `-b` option to tell lsof to avoid using kernel functions that would block. Some cautions apply.

First, using this option usually requires that your system supply alternate device numbers **in** place of the device numbers that lsof would normally obtain with the `lstat(2)` and `stat(2)` kernel functions. See the ALTERNATE DEVICE NUMBERS section **for** more information on alternate device numbers.

Second, you can't specify names **for** lsof to locate unless they're file system names. This is because lsof needs to know the device and inode numbers of files listed with names **in** the lsof options, and the `-b` option prevents lsof from obtaining them. Moreover, since lsof only has device numbers **for** the file systems that have alternates, its ability to locate files on file systems depends completely on the availability and accuracy of the alternates. If no alternates are available, or **if** they're incorrect, lsof won't be able to locate files on the named file systems.

Third, **if** the names of your file system directories that lsof obtains from your system's mount table are symbolic links, lsof won't be able to resolve the links. This is because the `-b` option causes lsof to avoid the kernel `readlink(2)` **function** it uses to resolve symbolic links.

Finally, using the `-b` option causes lsof to issue warning messages when it needs to use the kernel functions that the `-b` option directs it to avoid. You can suppress these messages by specifying the `-w` option, but **if** you **do**, you won't see the alternate device numbers reported **in** the warning messages.

ALTERNATE DEVICE NUMBERS

On some dialects, when lsof has to **break** a block because it can't get information about a mounted file system via the `lstat(2)` and `stat(2)` kernel functions, or because you specified the `-b` option, lsof can obtain some of the information it needs - the device number and possibly the file system **type** - from the system mount table. When that is possible, lsof will report the device number it obtained. (You can suppress the report by specifying the `-w` option.)

You can assist this process **if** your mount table is supported with

an /etc/mstab or /etc/mnttab file that contains an options field by adding a ``dev=xxxx'' field **for** mount points that **do** not have one **in** their options strings. Note: you must be able to edit the file - i.e., some mount tables like recent Solaris /etc/mnttab or Linux /proc/mounts are **read-only** and can't be modified.

You may also be able to supply device numbers using the +m and +m m options, provided they are supported by your dialect. Check the output of lsof's -h or -? options to see **if** the +m and +m m options are available.

The ``xxxx'' portion of the field is the hexadecimal value of the file system's device number. (Consult the st_dev field of the output of the lstat(2) and stat(2) functions **for** the appropriate values **for** your file systems.) Here's an example from a Sun Solaris 2.6 /etc/mnttab **for** a file system remotely mounted via NFS:

```
nfs ignore,noquota,dev=2a40001
```

There's an advantage to having ``dev=xxxx'' entries **in** your mount table file, especially **for** file systems that are mounted from remote NFS servers. When a remote server crashes and you want to identify its users by running lsof on one of its clients, lsof probably won't be able to get output from the lstat(2) and stat(2) functions **for** the file system. If it can obtain the file system's device number from the mount table, it will be able to display the files open on the crashed NFS server.

Some dialects that **do** not use an ASCII /etc/mstab or /etc/mnttab file **for** the mount table may still provide an alternative device number **in** their internal mount tables. This includes AIX, Apple Darwin, FreeBSD, NetBSD, OpenBSD, and Tru64 UNIX. Lsof knows how to obtain the alternative device number **for** these dialects and uses it when its attempt to lstat(2) or stat(2) the file system is blocked.

If you're not sure your dialect supplies alternate device numbers **for** file systems from its mount table, use this lsof incantation to see **if** it reports any alternate device numbers:

```
lsof -b
```

Look **for** standard error file warning messages that begin ``assuming "dev=xxxx" from ...''.

KERNEL NAME CACHE

Lsof is able to examine the kernel's name cache or use other kernel facilities (e.g., the ADVFS 4.x tag_to_path() **function** under Tru64 UNIX) on some dialects **for** most file system types, excluding AFS, and extract recently used path name components

from it. (AFS file system path lookups don't use the kernel's name cache; some Solaris VxFS file system operations apparently don't use it, either.)

Lsof reports the **complete** paths it finds **in** the NAME column. If lsof can't report all components **in** a path, it reports **in** the NAME column the file system name, followed by a space, two `-' characters, another space, and the name components it has located, separated by the `/' character.

When lsof is run **in** repeat mode - i.e., with the -r option specified - the extent to which it can report path name components **for** the same file may vary from cycle to cycle. That's because other running processes can cause the kernel to remove entries from its name cache and replace them with others.

Lsof's use of the kernel name cache to identify the paths of files can lead it to report incorrect components under some circumstances. This can happen when the kernel name cache uses device and node number as a key (e.g., SCO OpenServer) and a key on a rapidly changing file system is reused. If the UNIX dialect's kernel doesn't purge the name cache entry **for** a file when it is unlinked, lsof may find a reference to the wrong entry **in** the cache. The lsof FAQ (The FAQ section gives its location.) has more information on this situation.

Lsof can report path name components **for** these dialects:

- FreeBSD
- HP-UX
- Linux
- NetBSD
- NEXTSTEP
- OpenBSD
- OPENSTEP
- SCO OpenServer
- SCO|Caldera UnixWare
- Solaris
- Tru64 UNIX

Lsof can't report path name components **for** these dialects:

- AIX

If you want to know why lsof can't report path name components **for** some dialects, see the lsof FAQ (The FAQ section gives its location.)

DEVICE CACHE FILE

Examining all members of the /dev (or /devices) node tree with stat(2) functions can be time consuming. What's more, the

information that lsof needs - device number, inode number, and path - rarely changes.

Consequently, lsof normally maintains an ASCII text file of cached /dev (or /devices) information (exception: the /proc-based Linux lsof where it's not needed.) The **local** system administrator who builds lsof can control the way the device cache file path is formed, selecting from these options:

- Path from the -D option;
- Path from an environment variable;
- System-wide path;
- Personal path (the default);
- Personal path, modified by an environment variable.

Consult the output of the -h, -D? , or -? **help** options **for** the current state of device cache support. The **help** output lists the default **read**-mode device cache file path that is **in** effect **for** the current invocation of lsof. The -D? option output lists the **read**-only and write device cache file paths, the names of any applicable environment variables, and the personal device cache path format.

Lsof can detect that the current device cache file has been accidentally or maliciously modified by integrity checks, including the computation and verification of a sixteen bit Cyclic Redundancy Check (CRC) sum on the file's contents. When lsof senses something wrong with the file, it issues a warning and attempts to remove the current cache file and create a new copy, but only to a path that the process can legitimately write.

The path from which a lsof process may attempt to **read** a device cache file may not be the same as the path to which it can legitimately write. Thus when lsof senses that it needs to update the device cache file, it may choose a different path **for** writing it from the path from which it **read** an incorrect or outdated version.

If available, the -Dr option will inhibit the writing of a new device cache file. (It's always available when specified without a path name argument.)

When a new device is added to the system, the device cache file may need to be recreated. Since lsof compares the mtime of the device cache file with the mtime and ctime of the /dev (or /devices) directory, it usually detects that a new device has been added; **in** that **case** lsof issues a warning message and attempts to rebuild the device cache file.

Whenever lsof writes a device cache file, it sets its ownership

to the real UID of the executing process, and its permission modes to 0600, this restricting its reading and writing to the file's owner.

LSOF PERMISSIONS THAT AFFECT DEVICE CACHE FILE ACCESS

Two permissions of the lsof executable affect its ability to access device cache files. The permissions are **set** by the **local** system administrator when lsof is installed.

The first and rarer permission is setuid-root. It comes into effect when lsof is executed; its effective UID is **then** root, **while** its real (i.e., that of the logged-on user) UID is not. The lsof distribution recommends that versions **for** these dialects run setuid-root.

HP-UX 11.11 and 11.23
Linux

The second and more common permission is setgid. It comes into effect when the effective group IDentification number (GID) of the lsof process is **set** to one that can access kernel memory devices - e.g., ``kmem'', ``sys'', or ``system''.

An lsof process that has setgid permission usually surrenders the permission after it has accessed the kernel memory devices. When it does that, lsof can allow more liberal device cache path formations. The lsof distribution recommends that versions **for** these dialects run setgid and be allowed to surrender setgid permission.

AIX 5.[12] and 5.3-ML1
Apple Darwin 7.x Power Macintosh systems
FreeBSD 4.x, 4.1x, 5.x and [6789].x **for** x86-based systems
FreeBSD 5.x, [6789].x and 1[012].8for Alpha, AMD64 and Sparc64
based systems
HP-UX 11.00
NetBSD 1.[456], 2.x and 3.x **for** Alpha, x86, and SPARC-based
systems
NEXTSTEP 3.[13] **for** NEXTSTEP architectures
OpenBSD 2.[89] and 3.[0-9] **for** x86-based systems
OPENSTEP 4.x
SCO OpenServer Release 5.0.6 **for** x86-based systems
SCO|Caldera UnixWare 7.1.4 **for** x86-based systems
Solaris 2.6, 8, 9 and 10
Tru64 UNIX 5.1

(Note: lsof **for** AIX 5L and above needs setuid-root permission **if** its -X option is used.)

Lsof **for** these dialects does not support a device cache, so the permissions given to the executable don't apply to the device

cache file.

Linux

DEVICE CACHE FILE PATH FROM THE -D OPTION

The -D option provides limited means **for** specifying the device cache file path. Its ? **function** will report the **read**-only and write device cache file paths that lsof will use.

When the -D b, r, and u functions are available, you can use them to request that the cache file be built **in** a specific location (b[path]); **read** but not rebuilt (r[path]); or **read** and rebuilt (u[path]). The b, r, and u functions are restricted under some conditions. They are restricted when the lsof process is setuid-root. The path specified with the r **function** is always **read**-only, even when it is available.

The b, r, and u functions are also restricted when the lsof process runs setgid and lsof doesn't surrender the setgid permission. (See the LSOF PERMISSIONS THAT AFFECT DEVICE CACHE FILE ACCESS section **for** a list of implementations that normally don't surrender their setgid permission.)

A further -D **function**, i (**for** ignore), is always available.

When available, the b **function** tells lsof to **read** device information from the kernel with the stat(2) **function** and build a device cache file at the indicated path.

When available, the r **function** tells lsof to **read** the device cache file, but not update it. When a path argument accompanies -Dr, it names the device cache file path. The r **function** is always available when it is specified without a path name argument. If lsof is not running setuid-root and surrenders its setgid permission, a path name argument may accompany the r function.

When available, the u **function** tells lsof to attempt to **read** and use the device cache file. If it can't **read** the file, or **if** it finds the contents of the file incorrect or outdated, it will **read** information from the kernel, and attempt to write an updated version of the device cache file, but only to a path it considers legitimate **for** the lsof process effective and real UIDs.

DEVICE CACHE PATH FROM AN ENVIRONMENT VARIABLE

Lsof's second choice **for** the device cache file is the contents of the LSOFDEVCACHE environment variable. It avoids this choice **if** the lsof process is setuid-root, or the real UID of the process is root.

A further restriction applies to a device cache file path taken from the LSOFDEVCACHE environment variable: lsof will not write a

device cache file to the path **if** the lsof process doesn't surrender its setgid permission. (See the LSOF PERMISSIONS THAT AFFECT DEVICE CACHE FILE ACCESS section **for** information on implementations that don't surrender their setgid permission.)

The **local** system administrator can disable the use of the LSOFDEVCACHE environment variable or change its name when building lsof. Consult the output of -D? **for** the environment variable's name.

SYSTEM-WIDE DEVICE CACHE PATH

The **local** system administrator may choose to have a system-wide device cache file when building lsof. That file will generally be constructed by a special system administration procedure when the system is booted or when the contents of /dev or /devices) changes. If defined, it is lsof's third device cache file path choice.

You can tell that a system-wide device cache file is **in** effect **for** your **local** installation by examining the lsof **help** option output - i.e., the output from the -h or -? option.

Lsof will never write to the system-wide device cache file path by default. It must be explicitly named with a -D **function in** a root-owned procedure. Once the file has been written, the procedure must change its permission modes to 0644 (owner-**read** and owner-**write**, group-**read**, and other-**read**).

PERSONAL DEVICE CACHE PATH (DEFAULT)

The default device cache file path of the lsof distribution is one recorded **in** the home directory of the real UID that executes lsof. Added to the home directory is a second path component of the form .lsof_hostname.

This is lsof's fourth device cache file path choice, and is usually the default. If a system-wide device cache file path was defined when lsof was built, this fourth choice will be applied when lsof can't find the system-wide device cache file. This is the only time lsof uses two paths when reading the device cache file.

The hostname part of the second component is the base name of the executing host, as returned by gethostname(2). The base name is defined to be the characters preceding the first `.' **in** the gethostname(2) output, or all the gethostname(2) output **if** it contains no `.'.

The device cache file belongs to the user ID and is readable and writable by the user ID alone - i.e., its modes are 0600. Each distinct real user ID on a given host that executes lsof has a distinct device cache file. The hostname part of the path distinguishes device cache files **in** an NFS-mounted home directory

into which device cache files are written from several different hosts.

The personal device cache file path formed by this method represents a device cache file that lsof will attempt to **read**, and will attempt to write should it not exist or should its contents be incorrect or outdated.

The **-Dr** option without a path name argument will inhibit the writing of a new device cache file.

The **-D?** option will list the format specification **for** constructing the personal device cache file. The conversions used **in** the format specification are described **in** the 00DCACHE file of the lsof distribution.

MODIFIED PERSONAL DEVICE CACHE PATH

If this option is defined by the **local** system administrator when lsof is built, the LSOFPERSDCPATH environment variable contents may be used to add a component of the personal device cache file path.

The LSOFPERSDCPATH variable contents are inserted **in** the path at the place marked by the **local** system administrator with the ``%p'' conversion **in** the HASPERSDC format specification of the dialect's machine.h header file. (It's placed right after the home directory **in** the default lsof distribution.)

Thus, **for** example, **if** LSOFPERSDCPATH contains ``LSOF'', the home directory is ``/Homes/abe'', the host name is ``lsof.itap.purdue.edu'', and the HASPERSDC format is the default (``%h/%p.lsof_%L''), the modified personal device cache file path is:

```
/Homes/abe/LSOF/.lsof_vic
```

The LSOFPERSDCPATH environment variable is ignored when the lsof process is setuid-root or when the real UID of the process is root.

Lsof will not write to a modified personal device cache file path **if** the lsof process doesn't surrender setgid permission. (See the LSOF PERMISSIONS THAT AFFECT DEVICE CACHE FILE ACCESS section **for** a list of implementations that normally don't surrender their setgid permission.)

If, **for** example, you want to create a sub-directory of personal device cache file paths by using the LSOFPERSDCPATH environment variable to name it, and lsof doesn't surrender its setgid permission, you will have to allow lsof to create device cache files at the standard personal path and move them to your

subdirectory with shell commands.

The **local** system administrator may: disable this option when **lsof** is built; change the name of the environment variable from **LSOFPERSDCPATH** to something **else**; change the **HASPERSDC** format to include the personal path component **in** another place; or exclude the personal path component entirely. Consult the output of the **-D?** option **for** the environment variable's name and the **HASPERSDC** format specification.

DIAGNOSTICS

Errors are identified with messages on the standard error file.

Lsof returns a one (1) **if** any error was detected, including the failure to locate **command** names, file names, Internet addresses or files, **login** names, NFS files, PIDs, PGIDs, or UIDs it was asked to list. If the **-V** option is specified, **lsof** will indicate the search items it failed to list.

It returns a zero (0) **if** no errors were detected and **if** it was able to list some information about all the specified search arguments.

When **lsof** cannot open access to **/dev** (or **/devices**) or one of its subdirectories, or get information on a file **in** them with **stat(2)**, it issues a warning message and continues. That **lsof** will issue warning messages about inaccessible files **in** **/dev** (or **/devices**) is indicated **in** its **help** output - requested with the **-h** or **>B -?** options - with the message:

Inaccessible /dev warnings are enabled.

The warning message may be suppressed with the **-w** option. It may also have been suppressed by the system administrator when **lsof** was compiled by the setting of the **WARNDEVACCESS** definition. In this **case**, the output from the **help** options will include the message:

Inaccessible /dev warnings are disabled.

Inaccessible device warning messages usually disappear after **lsof** has created a working device cache file.

EXAMPLES

For a more extensive **set** of examples, documented more fully, see the **00QUICKSTART** file of the **lsof** distribution.

To list all open files, use:

```
lsof
```

To list all open Internet, x.25 (HP-UX), and UNIX domain files,

use:

```
lsof -i -U
```

To list all open IPv4 network files **in** use by the process whose PID is 1234, use:

```
lsof -i 4 -a -p 1234
```

Presuming the UNIX dialect supports IPv6, to list only open IPv6 network files, use:

```
lsof -i 6
```

To list all files using any protocol on ports 513, 514, or 515 of host wonderland.cc.purdue.edu, use:

```
lsof -i @wonderland.cc.purdue.edu:513-515
```

To list all files using any protocol on any port of mace.cc.purdue.edu (cc.purdue.edu is the default domain), use:

```
lsof -i @mace
```

To list all open files **for login** name ``abe'', or user ID 1234, or process 456, or process 123, or process 789, use:

```
lsof -p 456,123,789 -u 1234,abe
```

To list all open files on device /dev/hd4, use:

```
lsof /dev/hd4
```

To find the process that has /u/abe/foo open, use:

```
lsof /u/abe/foo
```

To send a SIGHUP to the processes that have /u/abe/bar open, use:

```
kill -HUP `lsof -t /u/abe/bar`
```

To find any open file, including an open UNIX domain socket file, with the name /dev/log, use:

```
lsof /dev/log
```

To find processes with open files on the NFS file system named /nfs/mount/point whose server is inaccessible, and presuming your mount table supplies the device number **for** /nfs/mount/point, use:

```
lsof -b /nfs/mount/point
```

To **do** the preceding search with warning messages suppressed, use:

```
lsof -bw /nfs/mount/point
```

To ignore the device cache file, use:

```
lsof -Di
```

To obtain PID and **command** name field output **for** each process, file descriptor, file device number, and file inode number **for** each file of each process, use:

```
lsof -FpcfDi
```

To list the files at descriptors 1 and 3 of every process running the **lsof command for login ID** ``abe'' every 10 seconds, use:

```
lsof -c lsof -a -d 1 -d 3 -u abe -r10
```

To list the current working directory of processes running a **command** that is exactly four characters long and has an 'o' or '0' **in** character three, use this regular expression form of the -c c option:

```
lsof -c /^..o.$/i -a -d cwd
```

To find an IP version 4 socket file by its associated numeric dot-form address, use:

```
lsof -i@128.210.15.17
```

To find an IP version 6 socket file (when the UNIX dialect supports IPv6) by its associated numeric colon-form address, use:

```
lsof -i@[0:1:2:3:4:5:6:7]
```

To find an IP version 6 socket file (when the UNIX dialect supports IPv6) by an associated numeric colon-form address that has a run of zeroes **in** it - e.g., the loop-back address - use:

```
lsof -i@[::1]
```

To obtain a repeat mode marker line that contains the current time, use:

```
lsof -rm====%T====
```

To add spaces to the previous marker line, use:


```
lsof -r "m====_T_===="
```

BUGS

Since lsof reads kernel memory **in** its search **for** open files, rapid changes **in** kernel memory may produce unpredictable results.

When a file has multiple record locks, the lock status character (following the file descriptor) is derived from a **test** of the first lock structure, not from any combination of the individual record locks that might be described by multiple lock structures.

Lsof can't search **for** files with restrictive access permissions by name unless it is installed with root **set**-UID permission. Otherwise it is limited to searching **for** files to which its user or its **set**-GID group (**if** any) has access permission.

The display of the destination address of a raw socket (e.g., **for** ping) depends on the UNIX operating system. Some dialects store the destination address **in** the raw socket's protocol control block, some **do** not.

Lsof can't always represent Solaris device numbers **in** the same way that ls(1) does. For example, the major and minor device numbers that the lstat(2) and stat(2) functions report **for** the directory on which CD-ROM files are mounted (typically /cdrom) are not the same as the ones that it reports **for** the device on which CD-ROM files are mounted (typically /dev/sr0). (Lsof reports the directory numbers.)

The support **for** /proc file systems is available only **for** BSD and Tru64 UNIX dialects, Linux, and dialects derived from SYSV R4 - e.g., FreeBSD, NetBSD, OpenBSD, Solaris, UnixWare.

Some /proc file items - device number, inode number, and file size - are unavailable **in** some dialects. Searching **for** files **in** a /proc file system may require that the full path name be specified.

No text (txt) file descriptors are displayed **for** Linux processes. All entries **for** files other than the current working directory, the root directory, and numerical file descriptors are labeled mem descriptors.

Lsof can't search **for** Tru64 UNIX named pipes by name, because their kernel implementation of lstat(2) returns an improper device number **for** a named pipe.

Lsof can't report fully or correctly on HP-UX 9.01, 10.20, and 11.00 locks because of insufficient access to kernel data or errors **in** the kernel data. See the lsof FAQ (The FAQ section

gives its location.) **for** details.

The AIX SMT file **type** is a fabrication. It's made up **for** file structures whose **type** (15) isn't defined **in** the AIX /usr/include/sys/file.h header file. One way to create such file structures is to run X clients with the DISPLAY variable **set** to ``:0.0''.

The +|-f[cfgGn] option is not supported under /proc-based Linux lsof, because it doesn't **read** kernel structures from kernel memory.

ENVIRONMENT

Lsof may access these environment variables.

LANG defines a language locale. See setlocale(3) **for** the names of other variables that can be used **in** place of LANG - e.g., LC_ALL, LC_TYPE, etc.

LSOFDEVCACHE

defines the path to a device cache file. See the DEVICE CACHE PATH FROM AN ENVIRONMENT VARIABLE section **for** more information.

LSOFPERSDCPATH

defines the middle component of a modified personal device cache file path. See the MODIFIED PERSONAL DEVICE CACHE PATH section **for** more information.

FAQ

Frequently-asked questions and their answers (an FAQ) are available **in** the 00FAQ file of the lsof distribution.

That file is also available via anonymous ftp from lsof.itap.purdue.edu at pub/tools/unix/lsofFAQ. The URL is:

<ftp://lsof.itap.purdue.edu/pub/tools/unix/lsof/FAQ>

FILES

/dev/kmem
kernel virtual memory device

/dev/mem
physical memory device

/dev/swap
system paging device

.lsof_hostname
lsof's device cache file (The suffix, hostname, is the first component of the host's name returned by gethostname(2).)

AUTHORS

Lsof was written by Victor A. Abell <abe@purdue.edu> of Purdue University. Many others have contributed to lsof. They're listed **in** the 00CREDITS file of the lsof distribution.

DISTRIBUTION

The latest distribution of lsof is available via anonymous ftp from the host lsof.itap.purdue.edu. You'll find the lsof distribution **in** the pub/tools/unix/lsof directory.

You can also use this URL:

`ftp://lsof.itap.purdue.edu/pub/tools/unix/lsof`

Lsof is also mirrored elsewhere. When you access lsof.itap.purdue.edu and change to its pub/tools/unix/lsof directory, you'll be given a list of some mirror sites. The pub/tools/unix/lsof directory also contains a more **complete** list **in** its mirrors file. Use mirrors with caution - not all mirrors always have the latest lsof revision.

Some pre-compiled Lsof executables are available on lsof.itap.purdue.edu, but their use is discouraged - it's better that you build your own from the sources. If you feel you must use a pre-compiled executable, please **read** the cautions that appear **in** the README files of the pub/tools/unix/lsof/binaries subdirectories and **in** the 00* files of the distribution.

More information on the lsof distribution can be found **in** its README.lsof_<version> file. If you intend to get the lsof distribution and build it, please **read** README.lsof_<version> and the other 00* files of the distribution before sending questions to the author.

SEE ALSO

Not all the following manual pages may exist **in** every UNIX dialect to which lsof has been ported.

access(2), **awk**(1), crash(1), fattach(3C), ff(1), fstat(8), fuser(1), gethostname(2), isprint(3), **kill**(1), localtime(3), lstat(2), modload(8), mount(8), netstat(1), ofiles(8L), perl(1), **ps**(1), readlink(2), setlocale(3), stat(2), strftime(3), time(2), uname(1).

COLOPHON

This page is part of the lsof (LiSt Open Files) project. Information about the project can be found at <http://people.freebsd.org/~abe/>. If you have a bug report **for** this manual page, send it to abe@purdue.edu. This page was obtained from the tarball lsof_4.91_src.tar fetched from <ftp://ftp.fu-berlin.de/pub/unix/tools/lsof/lsof.tar.gz> on 2024-06-14. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or

improvements to the information in this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

Revision-4.91

LSOF(8)

3.5 objdump: Display Information From Object Files

NAME

objdump - display information from object files

SYNOPSIS

```
objdump [-a|--archive-headers]
         [-b bfdname|--target=bfdname]
         [-C|--demangle[=style] ]
         [-d|--disassemble[=symbol]]
         [-D|--disassemble-all]
         [-z|--disassemble-zeroes]
         [-EB|--endian={big | little }]
         [-f|--file-headers]
         [-F|--file-offsets]
         [--file-start-context]
         [-g|--debugging]
         [-e|--debugging-tags]
         [-h|--section-headers|--headers]
         [-i|--info]
         [-j section|--section=section]
         [-l|--line-numbers]
         [-S|--source]
         [--source-comment[=text]]
         [-m machine|--architecture=machine]
         [-M options|--disassembler-options=options]
         [-p|--private-headers]
         [-P options|--private=options]
         [-r|--reloc]
         [-R|--dynamic-reloc]
         [-s|--full-contents]
         [-Z|--decompress]
         [-W[llIaprmfFsoORtUuTgAck]|
         --dwarf[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=fr
         [-WK|--dwarf=follow-links]
         [-WN|--dwarf=no-follow-links]
         [-wD|--dwarf=use-debuginfod]
         [-wE|--dwarf=do-not-use-debuginfod]
         [-L|--process-links]
         [--ctf=section]
         [--sframe=section]
         [-G|--stabs]
         [-t|--syms]
         [-T|--dynamic-syms]
```

```

[-x|--all-headers]
[-w|--wide]
[--start-address=address]
[--stop-address=address]
[--no-addresses]
[--prefix-addresses]
[--[no-]show-raw-insn]
[--adjust-vma=offset]
[--show-all-symbols]
[--dwarf-depth=n]
[--dwarf-start=n]
[--ctf-parent=section]
[--no-recurse-limit|--recurse-limit]
[--special-syms]
[--prefix=prefix]
[--prefix-strip=level]
[--insn-width=width]
[--visualize-jumps[=color|=extended-color|=off]]
[--disassembler-color=[off|terminal|on|extended]]
[-U method] [--unicode=method]
[-V|--version]
[-H|--help]
objfile...

```

DESCRIPTION

objdump displays information about one or more object files. The options control what particular information to display. This information is mostly useful to programmers who are working on the compilation tools, as opposed to programmers who just want their program to compile and work.

objfile... are the object files to be examined. When you specify archives, objdump shows information on each of the member object files.

OPTIONS

The long and short forms of options, shown here as alternatives, are equivalent. At least one option from the list -a,-d,-D,-e,-f,-g,-G,-h,-H,-p,-P,-r,-R,-s,-S,-t,-T,-V,-x must be given.

-a

--archive-header

If any of the objfile files are archives, display the archive header information (**in** a format similar to `ls -l`). Besides the information you could list with `ar tv`, `objdump -a` shows the object file format of each archive member.

--adjust-vma=offset

When dumping information, first add offset to all the section addresses. This is useful **if** the section addresses **do** not correspond to the symbol table, which can happen when putting

sections at particular addresses when using a format which can not represent section addresses, such as a.out.

-b bfdname

--target=bfdname

Specify that the object-code format **for** the object files is bfdname. This option may not be necessary; objdump can automatically recognize many formats.

For example,

```
objdump -b oasys -m vax -h fu.o
```

displays summary information from the section headers (-h) of fu.o, which is explicitly identified (-m) as a VAX object file **in** the format produced by Oasys compilers. You can list the formats available with the -i option.

-C

--demangle[=style]

Decode (demangle) low-level symbol names into user-level names. Besides removing any initial underscore prepended by the system, this makes C++ **function** names readable. Different compilers have different mangling styles. The optional demangling style argument can be used to choose an appropriate demangling style **for** your compiler.

--recurse-limit

--no-recurse-limit

--recursion-limit

--no-recursion-limit

Enables or disables a limit on the amount of recursion performed whilst demangling strings. Since the name mangling formats allow **for** an infinite level of recursion it is possible to create strings whose decoding will exhaust the amount of stack space available on the host machine, triggering a memory fault. The limit tries to prevent this from happening by restricting recursion to 2048 levels of nesting.

The default is **for** this limit to be enabled, but disabling it may be necessary **in** order to demangle truly complicated names. Note however that **if** the recursion limit is disabled **then** stack exhaustion is possible and any bug reports about such an event will be rejected.

-g

--debugging

Display debugging information. This attempts to parse STABS debugging format information stored **in** the file and print it

out using a C like syntax. If no STABS debugging was found this option falls back on the -W option to print any DWARF information **in** the file.

-e

--debugging-tags

Like -g, but the information is generated **in** a format compatible with ctags tool.

-d

--disassemble

--disassemble=symbol

Display the assembler mnemonics **for** the machine instructions from the input file. This option only disassembles those sections which are expected to contain instructions. If the optional symbol argument is given, **then** display the assembler mnemonics starting at symbol. If symbol is a **function** name **then** disassembly will stop at the end of the **function**, otherwise it will stop when the next symbol is encountered. If there are no matches **for** symbol **then** nothing will be displayed.

Note **if** the --dwarf=follow-links option is enabled **then** any symbol tables **in** linked debug info files will be **read in** and used when disassembling.

-D

--disassemble-all

Like -d, but disassemble the contents of all non-empty non-bss sections, not just those expected to contain instructions. -j may be used to **select** specific sections.

This option also has a subtle effect on the disassembly of instructions **in** code sections. When option -d is **in** effect objdump will assume that any symbols present **in** a code section occur on the boundary between instructions and it will refuse to disassemble across such a boundary. When option -D is **in** effect however this assumption is suppressed. This means that it is possible **for** the output of -d and -D to differ **if**, **for** example, data is stored **in** code sections.

If the target is an ARM architecture this switch also has the effect of forcing the disassembler to decode pieces of data found **in** code sections as **if** they were instructions.

Note **if** the --dwarf=follow-links option is enabled **then** any symbol tables **in** linked debug info files will be **read in** and used when disassembling.

--no-addresses

When disassembling, don't print addresses on each line or **for** symbols and relocation offsets. In combination with `--no-show-raw-insn` this may be useful **for** comparing compiler output.

`--prefix-addresses`

When disassembling, print the **complete** address on each line. This is the older disassembly format.

`-EB`

`-EL`

`--endian={big|little}`

Specify the endianness of the object files. This only affects disassembly. This can be useful when disassembling a file format which does not describe endianness information, such as S-records.

`-f`

`--file-headers`

Display summary information from the overall header of each of the objfile files.

`-F`

`--file-offsets`

When disassembling sections, whenever a symbol is displayed, also display the file offset of the region of data that is about to be dumped. If zeroes are being skipped, **then** when disassembly resumes, tell the user how many zeroes were skipped and the file offset of the location from where the disassembly resumes. When dumping sections, display the file offset of the location from where the dump starts.

`--file-start-context`

Specify that when displaying interlisted **source** code/disassembly (assumes `-S`) from a file that has not yet been displayed, extend the context to the start of the file.

`-h`

`--section-headers`

`--headers`

Display summary information from the section headers of the object file.

File segments may be relocated to nonstandard addresses, **for** example by using the `-Ttext`, `-Tdata`, or `-Tbss` options to `ld`. However, some object file formats, such as `a.out`, **do** not store the starting address of the file segments. In those situations, although `ld` relocates the sections correctly, using `objdump -h` to list the file section headers cannot show the correct addresses. Instead, it shows the usual

addresses, which are implicit **for** the target.

Note, **in** some cases it is possible **for** a section to have both the READONLY and the NOREAD attributes set. In such cases the NOREAD attribute takes precedence, but objdump will report both since the exact setting of the flag bits might be important.

-H

--help

Print a summary of the options to objdump and exit.

-i

--info

Display a list showing all architectures and object formats available **for** specification with -b or -m.

-j name

--section=name

Display information **for** section name. This option may be specified multiple times.

-L

--process-links

Display the contents of non-debug sections found **in** separate debuginfo files that are linked to the main file. This option automatically implies the -WK option, and only sections requested by other **command** line options will be displayed.

-l

--line-numbers

Label the display (using debugging information) with the filename and **source** line numbers corresponding to the object code or relocs shown. Only useful with -d, -D, or -r.

-m machine

--architecture=machine

Specify the architecture to use when disassembling object files. This can be useful when disassembling object files which **do** not describe architecture information, such as S-records. You can list the available architectures with the -i option.

For most architectures it is possible to supply an architecture name and a machine name, separated by a colon. For example foo:bar would refer to the bar machine **type in** the foo architecture. This can be helpful **if** objdump has been configured to support multiple architectures.

If the target is an ARM architecture **then** this switch has an additional effect. It restricts the disassembly to only those instructions supported by the architecture specified by machine. If it is necessary to use this switch because the input file does not contain any architecture information, but it is also desired to disassemble all the instructions use `-marm`.

`-M options`

`--disassembler-options=options`

Pass target specific information to the disassembler. Only supported on some targets. If it is necessary to specify more than one disassembler option **then** multiple `-M` options can be used or can be placed together into a comma separated list.

For ARC, `dsp` controls the printing of DSP instructions, `spfp` selects the printing of FPX single precision FP instructions, `dpfp` selects the printing of FPX double precision FP instructions, `quarkse_em` selects the printing of special QuarkSE-EM instructions, `fpuda` selects the printing of double precision assist instructions, `fpus` selects the printing of FPU single precision FP instructions, **while** `fpud` selects the printing of FPU double precision FP instructions. Additionally, one can choose to have all the immediates printed **in** hexadecimal using `hex`. By default, the short immediates are printed using the decimal representation, **while** the long immediate values are printed as hexadecimal.

`cpu=...` allows one to enforce a particular ISA when disassembling instructions, overriding the `-m` value or whatever is **in** the ELF file. This might be useful to **select** ARC EM or HS ISA, because architecture is same **for** those and disassembler relies on private ELF header data to decide **if** code is **for** EM or HS. This option might be specified multiple **times** - only the latest value will be used. Valid values are same as **for** the assembler `-mcpu=...` option.

If the target is an ARM architecture **then** this switch can be used to **select** which register name **set** is used during disassembler. Specifying `-M reg-names-std` (the default) will **select** the register names as used **in** ARM's instruction **set** documentation, but with register 13 called 'sp', register 14 called 'lr' and register 15 called 'pc'. Specifying `-M reg-names-apcs` will **select** the name **set** used by the ARM Procedure Call Standard, whilst specifying `-M reg-names-raw` will just use `r` followed by the register number.

There are also two variants on the APCS register naming scheme enabled by `-M reg-names-atpcs` and `-M reg-names-`

special-atpcs which use the ARM/Thumb Procedure Call Standard naming conventions. (Either with the normal register names or the special register names).

This option can also be used **for** ARM architectures to force the disassembler to interpret all instructions as Thumb instructions by using the switch `--disassembler-options=force-thumb`. This can be useful when attempting to disassemble thumb code produced by other compilers.

For AArch64 targets this switch can be used to **set** whether instructions are disassembled as the most general instruction using the `-M no-aliases` option or whether instruction notes should be generated as comments **in** the disassembly using `-M notes`.

For the x86, some of the options duplicate functions of the `-m` switch, but allow finer grained control.

"x86-64"

"i386"

"i8086"

Select disassembly **for** the given architecture.

"intel"

"att"

Select between intel syntax mode and AT&T syntax mode.

"amd64"

"intel64"

Select between AMD64 ISA and Intel64 ISA.

"intel-mnemonic"

"att-mnemonic"

Select between intel mnemonic mode and AT&T mnemonic mode. Note: "intel-mnemonic" implies "intel" and "att-mnemonic" implies "att".

"addr64"

"addr32"

"addr16"

"data32"

"data16"

Specify the default address size and operand size. These five options will be overridden **if** "x86-64", "i386" or "i8086" appear later **in** the option string.

"suffix"

When **in** AT&T mode and also **for** a limited **set** of

instructions when **in** Intel mode, instructs the disassembler to print a mnemonic suffix even when the suffix could be inferred by the operands or, **for** certain instructions, the execution mode's defaults.

For PowerPC, the **-M** argument **raw** selects disassembly of hardware insns rather than aliases. For example, you will see **"rlwinm"** rather than **"clrlwi"**, and **"addi"** rather than **"li"**. All of the **-m** arguments **for** gas that **select** a CPU are supported. These are: 403, 405, 440, 464, 476, 601, 603, 604, 620, 7400, 7410, 7450, 7455, 750cl, 821, 850, 860, a2, booke, booke32, cell, com, e200z2, e200z4, e300, e500, e500mc, e500mc64, e500x2, e5500, e6500, efs, power4, power5, power6, power7, power8, power9, power10, ppc, ppc32, ppc64, ppc64bridge, ppcps, pwr, pwr2, pwr4, pwr5, pwr5x, pwr6, pwr7, pwr8, pwr9, pwr10, pwrX, titan, vle, and future. 32 and 64 modify the default or a prior CPU selection, disabling and enabling 64-bit insns respectively. In addition, altivec, any, lsp, htm, vsx, spe and spe2 add capabilities to a previous or later CPU selection. any will disassemble any opcode known to binutils, but **in** cases where an opcode has two different meanings or different arguments, you may not see the disassembly you expect. If you disassemble without giving a CPU selection, a default will be chosen from information gleaned by BFD from the object files headers, but the result again may not be as you expect.

For MIPS, this option controls the printing of instruction mnemonic names and register names **in** disassembled instructions. Multiple selections from the following may be specified as a comma separated string, and invalid options are ignored:

"no-aliases"

Print the 'raw' instruction mnemonic instead of some pseudo instruction mnemonic. I.e., print 'daddu' or 'or' instead of 'move', 'sll' instead of 'nop', etc.

"msa"

Disassemble MSA instructions.

"virt"

Disassemble the virtualization ASE instructions.

"xpa"

Disassemble the eXtended Physical Address (XPA) ASE instructions.

"gpr-names=ABI"

Print GPR (general-purpose register) names as appropriate

for the specified ABI. By default, GPR names are selected according to the ABI of the binary being disassembled.

"fpr-names=ABI"

Print FPR (floating-point register) names as appropriate **for** the specified ABI. By default, FPR numbers are printed rather than names.

"cp0-names=ARCH"

Print CP0 (system control coprocessor; coprocessor 0) register names as appropriate **for** the CPU or architecture specified by ARCH. By default, CP0 register names are selected according to the architecture and CPU of the binary being disassembled.

"hwr-names=ARCH"

Print HWR (hardware register, used by the "rdhwr" instruction) names as appropriate **for** the CPU or architecture specified by ARCH. By default, HWR names are selected according to the architecture and CPU of the binary being disassembled.

"reg-names=ABI"

Print GPR and FPR names as appropriate **for** the selected ABI.

"reg-names=ARCH"

Print CPU-specific register names (CP0 register and HWR names) as appropriate **for** the selected CPU or architecture.

For any of the options listed above, ABI or ARCH may be specified as numeric to have numbers printed rather than names, **for** the selected types of registers. You can list the available values of ABI and ARCH using the **--help** option.

For VAX, you can specify **function** entry addresses with -M entry:0xf00ba. You can use this multiple **times** to properly disassemble VAX binary files that don't contain symbol tables (like ROM dumps). In these cases, the **function** entry mask would otherwise be decoded as VAX instructions, which would probably lead the rest of the **function** being wrongly disassembled.

-p

--private-headers

Print information that is specific to the object file format. The exact information printed depends upon the object file format. For some object file formats, no additional

information is printed.

-P options

--private=options

Print information that is specific to the object file format. The argument options is a comma separated list that depends on the format (the lists of options is displayed with the **help**).

For XCOFF, the available options are:

"header"
"aout"
"sections"
"syms"
"relocs"
"lineno,"
"loader"
"except"
"typchk"
"traceback"
"toc"
"ldinfo"

For PE, the available options are:

"header"
"sections"

Not all object formats support this option. In particular the ELF format does not use it.

-r

--reloc

Print the relocation entries of the file. If used with -d or -D, the relocations are printed interspersed with the disassembly.

-R

--dynamic-reloc

Print the dynamic relocation entries of the file. This is only meaningful **for** dynamic objects, such as certain types of shared libraries. As **for** -r, **if** used with -d or -D, the relocations are printed interspersed with the disassembly.

-S

--full-contents

Display the full contents of sections, often used **in** combination with -j to request specific sections. By default all non-empty non-bss sections are displayed. By default any

compressed section will be displayed **in** its compressed form. In order to see the contents **in** a decompressed form add the -Z option to the **command** line.

-S

--source

Display **source** code intermixed with disassembly, **if** possible. Implies -d.

--show-all-symbols

When disassembling, show all the symbols that match a given address, not just the first one.

--source-comment[=txt]

Like the -S option, but all **source** code lines are displayed with a prefix of txt. Typically txt will be a comment string which can be used to distinguish the assembler code from the **source** code. If txt is not provided **then** a default string of "#_" (**hash** followed by a space), will be used.

--prefix=prefix

Specify prefix to add to the absolute paths when used with -S.

--prefix-strip=level

Indicate how many initial directory names to strip off the hardcoded absolute paths. It has no effect without --prefix=prefix.

--show-raw-insn

When disassembling instructions, print the instruction **in** hex as well as **in** symbolic form. This is the default except when --prefix-addresses is used.

--no-show-raw-insn

When disassembling instructions, **do** not print the instruction bytes. This is the default when --prefix-addresses is used.

--insn-width=width

Display width bytes on a single line when disassembling instructions.

--visualize-jumps[=color|=extended-color|=off]

Visualize jumps that stay inside a **function** by drawing ASCII art between the start and target addresses. The optional =color argument adds color to the output using simple terminal colors. Alternatively the =extended-color argument will add color using 8bit colors, but these might not work on all terminals.

If it is necessary to disable the visualize-jumps option after it has previously been enabled **then** use visualize-jumps=off.

--disassembler-color=off
--disassembler-color=terminal
--disassembler-color=on|color|colour
--disassembler-color=extened|extended-color|extened-colour
Enables or disables the use of colored syntax highlighting **in** disassembly output. The default behaviour is determined via a configure time option. Note, not all architectures support colored syntax highlighting, and depending upon the terminal used, colored output may not actually be legible.

The on argument adds colors using simple terminal colors.

The terminal argument does the same, but only **if** the output device is a terminal.

The extended-color argument is similar to the on argument, but it uses 8-bit colors. These may not work on all terminals.

The off argument disables colored disassembly.

-W[llLiaprmfFsoORtUuTgAckK]

--dwarf[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=frames-inte

Displays the contents of the DWARF debug sections **in** the file, **if** any are present. Compressed debug sections are automatically decompressed (temporarily) before they are displayed. If one or more of the optional letters or words follows the switch **then** only those **type(s)** of data will be dumped. The letters and words refer to the following information:

"a"

"=abbrev"

Displays the contents of the .debug_abbrev section.

"A"

"=addr"

Displays the contents of the .debug_addr section.

"c"

"=cu_index"

Displays the contents of the .debug_cu_index and/or .debug_tu_index sections.

"f"

"=frames"

Display the raw contents of a `.debug_frame` section.

"F"

"=frames-interp"

Display the interpreted contents of a `.debug_frame` section.

"g"

"=gdb_index"

Displays the contents of the `.gdb_index` and/or `.debug_names` sections.

"i"

"=info"

Displays the contents of the `.debug_info` section. Note: the output from this option can also be restricted by the use of the `--dwarf-depth` and `--dwarf-start` options.

"k"

"=links"

Displays the contents of the `.gnu_debuglink`, `.gnu_debugaltlink` and `.debug_sup` sections, **if** any of them are present. Also displays any links to separate dwarf object files (dwo), **if** they are specified by the `DW_AT_GNU_dwo_name` or `DW_AT_dwo_name` attributes **in** the `.debug_info` section.

"K"

"=follow-links"

Display the contents of any selected debug sections that are found **in** linked, separate debug info file(s). This can result **in** multiple versions of the same debug section being displayed **if** it exists **in** more than one file.

In addition, when displaying DWARF attributes, **if** a form is found that references the separate debug info file, **then** the referenced contents will also be displayed.

Note - **in** some distributions this option is enabled by default. It can be disabled via the `N debug` option. The default can be chosen when configuring the binutils via the `--enable-follow-debug-links=yes` or `--enable-follow-debug-links=no` options. If these are not used **then** the default is to **enable** the following of debug links.

Note - **if** support **for** the debuginfod protocol was enabled when the binutils were built **then** this option will also include an attempt to contact any debuginfod servers mentioned **in** the `DEBUGINFOD_URLS` environment variable.

This could take some time to resolve. This behaviour can be disabled via the `=do-not-use-debuginfod` debug option.

"N"
"=no-follow-links"
Disables the following of links to separate debug info files.

"D"
"=use-debuginfod"
Enables contacting debuginfod servers **if** there is a need to follow debug links. This is the default behaviour.

"E"
"=do-not-use-debuginfod"
Disables contacting debuginfod servers when there is a need to follow debug links.

"l"
"=rawline"
Displays the contents of the `.debug_line` section **in** a raw format.

"L"
"=decodedline"
Displays the interpreted contents of the `.debug_line` section.

"m"
"=macro"
Displays the contents of the `.debug_macro` and/or `.debug_macinfo` sections.

"o"
"=loc"
Displays the contents of the `.debug_loc` and/or `.debug_loclists` sections.

"O"
"=str-offsets"
Displays the contents of the `.debug_str_offsets` section.

"p"
"=pubnames"
Displays the contents of the `.debug_pubnames` and/or `.debug_gnu_pubnames` sections.

"r"
"=aranges"
Displays the contents of the `.debug_aranges` section.

"R"
 "=Ranges"
 Displays the contents of the .debug_ranges and/or .debug_rnglists sections.

"s"
 "=str"
 Displays the contents of the .debug_str, .debug_line_str and/or .debug_str_offsets sections.

"t"
 "=pubtype"
 Displays the contents of the .debug_pubtypes and/or .debug_gnu_pubtypes sections.

"T"
 "=trace_aranges"
 Displays the contents of the .trace_aranges section.

"u"
 "=trace_abbrev"
 Displays the contents of the .trace_abbrev section.

"U"
 "=trace_info"
 Displays the contents of the .trace_info section.

Note: displaying the contents of .debug_static_funcs, .debug_static_vars and debug_weaknames sections is not currently supported.

--dwarf-depth=n
 Limit the dump of the ".debug_info" section to n children. This is only useful with --debug-dump=info. The default is to print all DIEs; the special value 0 **for** n will also have this effect.

With a non-zero value **for** n, DIEs at or deeper than n levels will not be printed. The range **for** n is zero-based.

--dwarf-start=n
 Print only DIEs beginning with the DIE numbered n. This is only useful with --debug-dump=info.

If specified, this option will suppress printing of any header information and all DIEs before the DIE numbered n. Only siblings and children of the specified DIE will be printed.

This can be used **in** conjunction with `--dwarf-depth`.

`--dwarf-check`

Enable additional checks **for** consistency of Dwarf information.

`--ctf[=section]`

Display the contents of the specified CTF section. CTF sections themselves contain many subsections, all of which are displayed **in** order.

By default, display the name of the section named `.ctf`, which is the name emitted by `ld`.

`--ctf-parent=member`

If the CTF section contains ambiguously-defined types, it will consist of an archive of many CTF dictionaries, all inheriting from one dictionary containing unambiguous types. This member is by default named `.ctf`, like the section containing it, but it is possible to change this name using the `"ctf_link_set_memb_name_changer"` **function** at link time. When looking at CTF archives that have been created by a linker that uses the name changer to rename the parent archive member, `--ctf-parent` can be used to specify the name used **for** the parent.

`--sframe[=section]`

Display the contents of the specified SFrame section.

By default, display the name of the section named `.sframe`, which is the name emitted by `ld`.

`-G`

`--stabs`

Display the full contents of any sections requested. Display the contents of the `.stab` and `.stab.index` and `.stab.excl` sections from an ELF file. This is only useful on systems (such as Solaris 2.0) **in** which `".stab"` debugging symbol-table entries are carried **in** an ELF section. In most other file formats, debugging symbol-table entries are interleaved with linkage symbols, and are visible **in** the `--syms` output.

`--start-address=address`

Start displaying data at the specified address. This affects the output of the `-d`, `-r` and `-s` options.

`--stop-address=address`

Stop displaying data at the specified address. This affects the output of the `-d`, `-r` and `-s` options.

```
-t
--syms
```

Print the symbol table entries of the file. This is similar to the information provided by the nm program, although the display format is different. The format of the output depends upon the format of the file being dumped, but there are two main types. One looks like this:

```
[ 4](sec 3)(fl 0x00)(ty 0)(scl 3) (nx 1) 0x00000000
.bss
[ 6](sec 1)(fl 0x00)(ty 0)(scl 2) (nx 0) 0x00000000
fred
```

where the number inside the square brackets is the number of the entry **in** the symbol table, the sec number is the section number, the fl value are the symbol's flag bits, the ty number is the symbol's **type**, the scl number is the symbol's storage class and the nx value is the number of auxiliary entries associated with the symbol. The last two fields are the symbol's value and its name.

The other common output format, usually seen with ELF based files, looks like this:

```
00000000 l d .bss 00000000 .bss
00000000 g .text 00000000 fred
```

Here the first number is the symbol's value (sometimes referred to as its address). The next field is actually a **set** of characters and spaces indicating the flag bits that are **set** on the symbol. These characters are described below. Next is the section with which the symbol is associated or ***ABS*** **if** the section is absolute (ie not connected with any section), or ***UND*** **if** the section is referenced **in** the file being dumped, but not defined there.

After the section name comes another field, a number, which **for** common symbols is the alignment and **for** other symbol is the size. Finally the symbol's name is displayed.

The flag characters are divided into 7 groups as follows:

```
"l"
"g"
"u"
```

"!" The symbol is a **local** (l), global (g), unique global (u), neither global nor **local** (a space) or both global and **local** (!). A symbol can be neither **local** or global **for** a variety of reasons, e.g., because it is used **for** debugging, but it is probably an indication of a bug **if**

it is ever both **local** and global. Unique global symbols are a GNU extension to the standard **set** of ELF symbol bindings. For such a symbol the dynamic linker will make sure that **in** the entire process there is just one symbol with this name and **type in** use.

"w" The symbol is weak (w) or strong (a space).

"C" The symbol denotes a constructor (C) or an ordinary symbol (a space).

"W" The symbol is a warning (W) or a normal symbol (a space). A warning symbol's name is a message to be displayed **if** the symbol following the warning symbol is ever referenced.

"I"

"i" The symbol is an indirect reference to another symbol (I), a **function** to be evaluated during reloc processing (i) or a normal symbol (a space).

"d"

"D" The symbol is a debugging symbol (d) or a dynamic symbol (D) or a normal symbol (a space).

"F"

"f"

"O" The symbol is the name of a **function** (F) or a file (f) or an object (O) or just a normal symbol (a space).

-T

--dynamic-syms

Print the dynamic symbol table entries of the file. This is only meaningful **for** dynamic objects, such as certain types of shared libraries. This is similar to the information provided by the nm program when given the -D (--dynamic) option.

The output format is similar to that produced by the --syms option, except that an extra field is inserted before the symbol's name, giving the version information associated with the symbol. If the version is the default version to be used when resolving unversioned references to the symbol **then** it's displayed as is, otherwise it's put into parentheses.

--special-syms

When displaying symbols include those which the target considers to be special **in** some way and which would not normally be of interest to the user.

-U [d|i|l|e|x|h]
 --unicode=[default|invalid|locale|escape|hex|highlight]
 Controls the display of UTF-8 encoded multibyte characters **in** strings. The default (--unicode=default) is to give them no special treatment. The --unicode=locale option displays the sequence **in** the current locale, which may or may not support them. The options --unicode=hex and --unicode=invalid display them as hex byte sequences enclosed by either angle brackets or curly braces.

The --unicode=escape option displays them as escape sequences (\uxxxx) and the --unicode=highlight option displays them as escape sequences highlighted **in** red (**if** supported by the output device). The colouring is intended to draw attention to the presence of unicode sequences where they might not be expected.

-V
 --version
 Print the version number of objdump and exit.

-x
 --all-headers
 Display all available header information, including the symbol table and relocation entries. Using -x is equivalent to specifying all of -a -f -h -p -r -t.

-w
 --wide
 Format some lines **for** output devices that have more than 80 columns. Also **do** not truncate symbol names when they are displayed.

-Z
 --disassemble-zeroes
 Normally the disassembly output will skip blocks of zeroes. This option directs the disassembler to disassemble those blocks, just like any other data.

-Z
 --decompress
 The -Z option is meant to be used **in** conjunction with the -s option. It instructs objdump to decompress any compressed sections before displaying their contents.

@file
 Read **command**-line options from file. The options **read** are inserted **in** place of the original @file option. If file does not exist, or cannot be **read**, **then** the option will be treated literally, and not removed.

Options **in** file are separated by whitespace. A whitespace character may be included **in** an option by surrounding the entire option **in** either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any such options will be processed recursively.

SEE ALSO

nm(1), readelf(1), and the Info entries **for** binutils.

COPYRIGHT

Copyright (c) 1991-2024 Free Software Foundation, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included **in** the section entitled "GNU_Free_Documentation_License".

COLOPHON

This page is part of the binutils (a collection of tools **for** working with executable binaries) project. Information about the project can be found at <http://www.gnu.org/software/binutils/>. If you have a bug report **for** this manual page, see http://sourceware.org/bugzilla/enter_bug.cgi?product=binutils. This page was obtained from the tarball binutils-2.42.tar.gz fetched from <https://ftp.gnu.org/gnu/binutils/> on 2024-06-14. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

binutils-2.42

2024-06-14

OBJDUMP(1)

3.6 readelf: Display Information On ELF Files

NAME

readelf - display information about ELF files

SYNOPSIS

```
readelf [-a|--all]
        [-h|--file-header]
        [-l|--program-headers|--segments]
        [-S|--section-headers|--sections]
        [-g|--section-groups]
        [-t|--section-details]
        [-e|--headers]
        [-s|--syms|--symbols]
        [--dyn-syms|--lto-syms]
```



```

[--sym-base=[0|8|10|16]]
[--demangle=style|--no-demangle]
[--quiet]
[--recurse-limit|--no-recurse-limit]
[-U method|--unicode=method]
[-X|--extra-sym-info|--no-extra-sym-info]
[-n|--notes]
[-r|--relocs]
[-u|--unwind]
[-d|--dynamic]
[-V|--version-info]
[-A|--arch-specific]
[-D|--use-dynamic]
[-L|--lint|--enable-checks]
[-x <number or name>|--hex-dump=<number or name>]
[-p <number or name>|--string-dump=<number or name>]
[-R <number or name>|--relocated-dump=<number or name>]
[-z|--decompress]
[-c|--archive-index]
[-w[LLiaprmmfFso0RtUuTgAck]|
  --debug-dump[=rawline,=decodedline,=info,=abbrev,=pubnames,=ranges,=macro,=frame]
[-wK|--debug-dump=follow-links]
[-wN|--debug-dump=no-follow-links]
[-wD|--debug-dump=use-debuginfod]
[-wE|--debug-dump=do-not-use-debuginfod]
[-P|--process-links]
[--dwarf-depth=n]
[--dwarf-start=n]
[--ctf=section]
[--ctf-parent=section]
[--ctf-symbols=section]
[--ctf-strings=section]
[--sframe=section]
[-I|--histogram]
[-v|--version]
[-W|--wide]
[-T|--silent-truncation]
[-H|--help]
elffile...

```

DESCRIPTION

`readelf` displays information about one or more ELF format object files. The options control what particular information to display.

`elffile...` are the object files to be examined. 32-bit and 64-bit ELF files are supported, as are archives containing ELF files.

This program performs a similar **function** to `objdump` but it goes into more detail and it exists independently of the BFD library,

so **if** there is a bug **in** BFD **then** readelf will not be affected.

OPTIONS

The long and short forms of options, shown here as alternatives, are equivalent. At least one option besides -v or -H must be given.

-a
--all
Equivalent to specifying --file-header, --program-headers, --sections, --symbols, --relocs, --dynamic, --notes, --version-info, --arch-specific, --unwind, --section-groups and --histogram.

Note - this option does not **enable** --use-dynamic itself, so **if** that option is not present on the **command** line **then** dynamic symbols and dynamic relocs will not be displayed.

-h
--file-header
Displays the information contained **in** the ELF header at the start of the file.

-l
--program-headers
--segments
Displays the information contained **in** the file's segment headers, **if** it has any.

--quiet
Suppress "no_symbols" diagnostic.

-S
--sections
--section-headers
Displays the information contained **in** the file's section headers, **if** it has any.

-g
--section-groups
Displays the information contained **in** the file's section groups, **if** it has any.

-t
--section-details
Displays the detailed section information. Implies -S.

-s
--symbols
--syms
Displays the entries **in** symbol table section of the file, **if**

it has one. If a symbol has version information associated with it **then** this is displayed as well. The version string is displayed as a suffix to the symbol name, preceded by an @ character. For example foo@VER_1. If the version is the default version to be used when resolving unversioned references to the symbol **then** it is displayed as a suffix preceded by two @ characters. For example foo@@VER_2.

--dyn-syms

Displays the entries **in** dynamic symbol table section of the file, **if** it has one. The output format is the same as the format used by the --syms option.

--lto-syms

Displays the contents of any LTO symbol tables **in** the file.

--sym-base=[0|8|10|16]

Forces the size field of the symbol table to use the given base. Any unrecognized options will be treated as 0.

--sym-base=0 represents the default and legacy behaviour.

This will output sizes as decimal **for** numbers less than 100000. For sizes 100000 and greater hexadecimal notation will be used with a 0x prefix. --sym-base=8 will give the symbol sizes **in** octal. --sym-base=10 will always give the symbol sizes **in** decimal. --sym-base=16 will always give the symbol sizes **in** hexadecimal with a 0x prefix.

-C

--demangle[=style]

Decode (demangle) low-level symbol names into user-level names. This makes C++ **function** names readable. Different compilers have different mangling styles. The optional demangling style argument can be used to choose an appropriate demangling style **for** your compiler.

--no-demangle

Do not demangle low-level symbol names. This is the default.

--recurse-limit

--no-recurse-limit

--recursion-limit

--no-recursion-limit

Enables or disables a limit on the amount of recursion performed whilst demangling strings. Since the name mangling formats allow **for** an infinite level of recursion it is possible to create strings whose decoding will exhaust the amount of stack space available on the host machine, triggering a memory fault. The limit tries to prevent this from happening by restricting recursion to 2048 levels of nesting.

The default is **for** this limit to be enabled, but disabling it may be necessary **in** order to demangle truly complicated names. Note however that **if** the recursion limit is disabled **then** stack exhaustion is possible and any bug reports about such an event will be rejected.

-U [d|i|l|e|x|h]

--unicode=[default|invalid|locale|escape|hex|highlight]

Controls the display of non-ASCII characters **in** identifier names. The default (**--unicode**=locale or **--unicode**=default) is to treat them as multibyte characters and display them **in** the current locale. All other versions of this option treat the bytes as UTF-8 encoded values and attempt to interpret them. If they cannot be interpreted or **if** the **--unicode**=invalid option is used **then** they are displayed as a sequence of hex bytes, enclosed **in** curly parenthesis characters.

Using the **--unicode**=escape option will display the characters as as unicode escape sequences (\uxxxx). Using the **--unicode**=hex will display the characters as hex byte sequences enclosed between angle brackets.

Using the **--unicode**=highlight will display the characters as unicode escape sequences but it will also highlighted them **in** red, assuming that colouring is supported by the output device. The colouring is intended to draw attention to the presence of unicode sequences when they might not be expected.

-X

--extra-sym-info

When displaying details of symbols, include extra information not normally presented. Currently this just adds the name of the section referenced by the symbol's index field, **if** there is one. In the future more information may be displayed when this option is enabled.

Enabling this option effectively enables the **--wide** option as well, at least when displaying symbol information.

--no-extra-sym-info

Disables the effect of the **--extra-sym-info** option. This is the default.

-e

--headers

Display all the headers **in** the file. Equivalent to **-h -l -S**.

```

-n
--notes
    Displays the contents of the NOTE segments and/or sections,
    if any.

-r
--relocs
    Displays the contents of the file's relocation section, if it
    has one.

-u
--unwind
    Displays the contents of the file's unwind section, if it has
    one. Only the unwind sections for IA64 ELF files, as well as
    ARM unwind tables (".ARM.exidx" / ".ARM.extab") are currently
    supported. If support is not yet implemented for your
    architecture you could try dumping the contents of the
    .eh_frames section using the --debug-dump=frames or
    --debug-dump=frames-interp options.

-d
--dynamic
    Displays the contents of the file's dynamic section, if it
    has one.

-V
--version-info
    Displays the contents of the version sections in the file, if
    they exist.

-A
--arch-specific
    Displays architecture-specific information in the file, if
    there is any.

-D
--use-dynamic
    When displaying symbols, this option makes readelf use the
    symbol hash tables in the file's dynamic section, rather than
    the symbol table sections.

    When displaying relocations, this option makes readelf
    display the dynamic relocations rather than the static
    relocations.

-L
--lint
--enable-checks
    Displays warning messages about possible problems with the
    file(s) being examined. If used on its own then all of the

```

contents of the file(s) will be examined. If used with one of the dumping options **then** the warning messages will only be produced **for** the things being displayed.

-x <number or name>

--hex-dump=<number or name>

Displays the contents of the indicated section as a hexadecimal bytes. A number identifies a particular section by index **in** the section table; any other string identifies all sections with that name **in** the object file.

-R <number or name>

--relocated-dump=<number or name>

Displays the contents of the indicated section as a hexadecimal bytes. A number identifies a particular section by index **in** the section table; any other string identifies all sections with that name **in** the object file. The contents of the section will be relocated before they are displayed.

-p <number or name>

--string-dump=<number or name>

Displays the contents of the indicated section as printable strings. A number identifies a particular section by index **in** the section table; any other string identifies all sections with that name **in** the object file.

-z

--decompress

Requests that the section(s) being dumped by x, R or p options are decompressed before being displayed. If the section(s) are not compressed **then** they are displayed as is.

-c

--archive-index

Displays the file symbol index information contained **in** the header part of binary archives. Performs the same **function** as the t **command** to ar, but without using the BFD library.

-w[llIaprmfFs0oRtUuTgAckK]

--debug-dump[=rawline,=decodedline,=info,=abbrev,=pubnames,=aranges,=macro,=frames,=frames

Displays the contents of the DWARF debug sections **in** the file, **if** any are present. Compressed debug sections are automatically decompressed (temporarily) before they are displayed. If one or more of the optional letters or words follows the switch **then** only those **type(s)** of data will be dumped. The letters and words refer to the following information:

"a"

"=abbrev"

```

    Displays the contents of the .debug_abbrev section.

"A"
"=addr"
    Displays the contents of the .debug_addr section.

"C"
"=cu_index"
    Displays the contents of the .debug_cu_index and/or
    .debug_tu_index sections.

"f"
"=frames"
    Display the raw contents of a .debug_frame section.

"F"
"=frames-interp"
    Display the interpreted contents of a .debug_frame
    section.

"g"
"=gdb_index"
    Displays the contents of the .gdb_index and/or
    .debug_names sections.

"i"
"=info"
    Displays the contents of the .debug_info section. Note:
    the output from this option can also be restricted by the
    use of the --dwarf-depth and --dwarf-start options.

"k"
"=links"
    Displays the contents of the .gnu_debuglink,
    .gnu_debugaltlink and .debug_sup sections, if any of them
    are present. Also displays any links to separate dwarf
    object files (dwo), if they are specified by the
    DW_AT_GNU_dwo_name or DW_AT_dwo_name attributes in the
    .debug_info section.

"K"
"=follow-links"
    Display the contents of any selected debug sections that
    are found in linked, separate debug info file(s). This
    can result in multiple versions of the same debug section
    being displayed if it exists in more than one file.

    In addition, when displaying DWARF attributes, if a form
    is found that references the separate debug info file,
    then the referenced contents will also be displayed.

```

Note - **in** some distributions this option is enabled by default. It can be disabled via the N debug option. The default can be chosen when configuring the binutils via the **--enable-follow-debug-links=yes** or **--enable-follow-debug-links=no** options. If these are not used **then** the default is to **enable** the following of debug links.

Note - **if** support **for** the debuginfod protocol was enabled when the binutils were built **then** this option will also include an attempt to contact any debuginfod servers mentioned **in** the DEBUGINFOD_URLS environment variable. This could take some time to resolve. This behaviour can be disabled via the **=do-not-use-debuginfod** debug option.

"N"

"=no-follow-links"

Disables the following of links to separate debug info files.

"D"

"=use-debuginfod"

Enables contacting debuginfod servers **if** there is a need to follow debug links. This is the default behaviour.

"E"

"=do-not-use-debuginfod"

Disables contacting debuginfod servers when there is a need to follow debug links.

"l"

"=rawline"

Displays the contents of the .debug_line section **in** a raw format.

"L"

"=decodedline"

Displays the interpreted contents of the .debug_line section.

"m"

"=macro"

Displays the contents of the .debug_macro and/or .debug_macroinfo sections.

"o"

"=loc"

Displays the contents of the .debug_loc and/or .debug_loclists sections.


```

"0"
"=str-offsets"
    Displays the contents of the .debug_str_offsets section.

"p"
"=pubnames"
    Displays the contents of the .debug_pubnames and/or
    .debug_gnu_pubnames sections.

"r"
"=ranges"
    Displays the contents of the .debug_ranges section.

"R"
"=Ranges"
    Displays the contents of the .debug_ranges and/or
    .debug_rnglists sections.

"s"
"=str"
    Displays the contents of the .debug_str, .debug_line_str
    and/or .debug_str_offsets sections.

"t"
"=pubtype"
    Displays the contents of the .debug_pubtypes and/or
    .debug_gnu_pubtypes sections.

"T"
"=trace_aranges"
    Displays the contents of the .trace_aranges section.

"u"
"=trace_abbrev"
    Displays the contents of the .trace_abbrev section.

"U"
"=trace_info"
    Displays the contents of the .trace_info section.

```

Note: displaying the contents of .debug_static_funcs, .debug_static_vars and debug_weaknames sections is not currently supported.

```

--dwarf-depth=n
    Limit the dump of the ".debug_info" section to n children.
    This is only useful with --debug-dump=info. The default is
    to print all DIEs; the special value 0 for n will also have
    this effect.

```

With a non-zero value **for** n, DIEs at or deeper than n levels will not be printed. The range **for** n is zero-based.

--dwarf-start=n

Print only DIEs beginning with the DIE numbered n. This is only useful with --debug-dump=info.

If specified, this option will suppress printing of any header information and all DIEs before the DIE numbered n. Only siblings and children of the specified DIE will be printed.

This can be used **in** conjunction with --dwarf-depth.

-P

--process-links

Display the contents of non-debug sections found **in** separate debuginfo files that are linked to the main file. This option automatically implies the -wK option, and only sections requested by other **command** line options will be displayed.

--ctf[=section]

Display the contents of the specified CTF section. CTF sections themselves contain many subsections, all of which are displayed **in** order.

By default, display the name of the section named .ctf, which is the name emitted by ld.

--ctf-parent=member

If the CTF section contains ambiguously-defined types, it will consist of an archive of many CTF dictionaries, all inheriting from one dictionary containing unambiguous types. This member is by default named .ctf, like the section containing it, but it is possible to change this name using the "ctf_link_set_memb_name_changer" **function** at link time. When looking at CTF archives that have been created by a linker that uses the name changer to rename the parent archive member, --ctf-parent can be used to specify the name used **for** the parent.

--ctf-symbols=section

--ctf-strings=section

Specify the name of another section from which the CTF file can inherit strings and symbols. By default, the ".symtab" and its linked string table are used.

If either of --ctf-symbols or --ctf-strings is specified, the

other must be specified as well.

-I

--histogram

Display a histogram of bucket list lengths when displaying the contents of the symbol tables.

-v

--version

Display the version number of readelf.

-W

--wide

Don't **break** output lines to fit into 80 columns. By default readelf breaks section header and segment listing lines **for** 64-bit ELF files, so that they fit into 80 columns. This option causes readelf to print each section header resp. each segment one a single line, which is far more readable on terminals wider than 80 columns.

-T

--silent-truncation

Normally when readelf is displaying a symbol name, and it has to truncate the name to fit into an 80 column display, it will add a suffix of "[...]" to the name. This **command** line option disables this behaviour, allowing 5 more characters of the name to be displayed and restoring the old behaviour of readelf (prior to release 2.35).

-H

--help

Display the **command**-line options understood by readelf.

@file

Read **command**-line options from file. The options **read** are inserted **in** place of the original @file option. If file does not exist, or cannot be **read**, **then** the option will be treated literally, and not removed.

Options **in** file are separated by whitespace. A whitespace character may be included **in** an option by surrounding the entire option **in** either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any such options will be processed recursively.

SEE ALSO

objdump(1), and the Info entries **for** binutils.

COPYRIGHT

Copyright (c) 1991-2024 Free Software Foundation, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included **in** the section entitled "GNU_Free_Documentation License".

COLOPHON

This page is part of the binutils (a collection of tools **for** working with executable binaries) project. Information about the project can be found at <http://www.gnu.org/software/binutils/>. If you have a bug report **for** this manual page, see http://sourceware.org/bugzilla/enter_bug.cgi?product=binutils. This page was obtained from the tarball binutils-2.42.tar.gz fetched from <https://ftp.gnu.org/gnu/binutils/> on 2024-06-14. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

binutils-2.42

2024-06-14

READEF(1)

3.7 nm: List Symbols From Object Files

NAME

nm - list symbols from object files

SYNOPSIS

```
nm [-A|-o|--print-file-name]
    [-a|--debug-syms]
    [-B|--format=bsd]
    [-C|--demangle[=style]]
    [-D|--dynamic]
    [-fformat|--format=format]
    [-g|--extern-only]
    [-h|--help]
    [--ifunc-chars=CHARS]
    [-j|--format=just-symbols]
    [-l|--line-numbers] [--inlines]
    [-n|-v|--numeric-sort]
    [-P|--portability]
    [-p|--no-sort]
    [-r|--reverse-sort]
    [-S|--print-size]
    [-s|--print-arnam]
    [-t radix|--radix=radix]
    [-u|--undefined-only]
    [-U|--defined-only]
```

```

[-V|--version]
[-W|--no-weak]
[-X 32_64]
[--no-demangle]
[--no-recurse-limit|--recurse-limit]]
[--plugin name]
[--size-sort]
[--special-syms]
[--synthetic]
[--target=bfdname]
[--unicode=method]
[--with-symbol-versions]
[--without-symbol-versions]
[objfile...]

```

DESCRIPTION

GNU nm lists the symbols from object files objfile.... If no object files are listed as arguments, nm assumes the file a.out.

For each symbol, nm shows:

- * The symbol value, **in** the radix selected by options (see below), or hexadecimal by default.
- * The symbol type. At least the following types are used; others are, as well, depending on the object file format. If lowercase, the symbol is usually **local**; **if** uppercase, the symbol is global (external). There are however a few lowercase symbols that are shown **for** special global symbols ("u", "v" and "w").

"A" The symbol's value is absolute, and will not be changed by further linking.

"B"

"b" The symbol is **in** the BSS data section. This section typically contains zero-initialized or uninitialized data, although the exact behavior is system dependent.

"C"

"c" The symbol is common. Common symbols are uninitialized data. When linking, multiple common symbols may appear with the same name. If the symbol is defined anywhere, the common symbols are treated as undefined references. The lower **case** c character is used when the symbol is **in** a special section **for** small commons.

"D"

"d" The symbol is **in** the initialized data section.

"G"

"g" The symbol is **in** an initialized data section **for** small objects. Some object file formats permit more efficient access to small data objects, such as a global int variable as opposed to a large global array.

"i" For PE format files this indicates that the symbol is **in** a section specific to the implementation of DLLs.

For ELF format files this indicates that the symbol is an indirect function. This is a GNU extension to the standard **set** of ELF symbol types. It indicates a symbol which **if** referenced by a relocation does not evaluate to its address, but instead must be invoked at runtime. The runtime execution will **then return** the value to be used **in** the relocation.

Note - the actual symbols display **for** GNU indirect symbols is controlled by the `--ifunc-chars` **command** line option. If this option has been provided **then** the first character **in** the string will be used **for** global indirect **function** symbols. If the string contains a second character **then** that will be used **for local** indirect **function** symbols.

"I" The symbol is an indirect reference to another symbol.

"N" The symbol is a debugging symbol.

"n" The symbol is **in** a non-data, non-code, non-debug **read-only** section.

"p" The symbol is **in** a stack unwind section.

"R"

"r" The symbol is **in** a **read** only data section.

"S"

"s" The symbol is **in** an uninitialized or zero-initialized data section **for** small objects.

"T"

"t" The symbol is **in** the text (code) section.

"U" The symbol is undefined.

"u" The symbol is a unique global symbol. This is a GNU extension to the standard **set** of ELF symbol bindings. For such a symbol the dynamic linker will make sure that **in** the entire process there is just one symbol with this name and **type in** use.

"V"

"v" The symbol is a weak object. When a weak defined symbol is linked with a normal defined symbol, the normal defined symbol is used with no error. When a weak undefined symbol is linked and the symbol is not defined, the value of the weak symbol becomes zero with no error. On some systems, uppercase indicates that a default value has been specified.

"W"

"w" The symbol is a weak symbol that has not been specifically tagged as a weak object symbol. When a weak defined symbol is linked with a normal defined symbol, the normal defined symbol is used with no error. When a weak undefined symbol is linked and the symbol is not defined, the value of the symbol is determined **in** a system-specific manner without error. On some systems, uppercase indicates that a default value has been specified.

"-" The symbol is a stabs symbol **in** an a.out object file. In this **case**, the next values printed are the stabs other field, the stabs desc field, and the stab type. Stabs symbols are used to hold debugging information.

"?" The symbol **type** is unknown, or object file format specific.

- * The symbol name. If a symbol has version information associated with it, **then** the version information is displayed as well. If the versioned symbol is undefined or hidden from linker, the version string is displayed as a suffix to the symbol name, preceded by an @ character. For example foo@VER_1. If the version is the default version to be used when resolving unversioned references to the symbol, **then** it is displayed as a suffix preceded by two @ characters. For example foo@@VER_2.

OPTIONS

The long and short forms of options, shown here as alternatives, are equivalent.

-A

-O

--print-file-name

Precede each symbol by the name of the input file (or archive member) **in** which it was found, rather than identifying the input file once only, before all of its symbols.

-a

```

--debug-syms
    Display all symbols, even debugger-only symbols; normally
    these are not listed.

-B The same as --format=bsd (for compatibility with the MIPS
    nm).

-C
--demangle[=style]
    Decode (demangle) low-level symbol names into user-level
    names. Besides removing any initial underscore prepended by
    the system, this makes C++ function names readable. Different
    compilers have different mangling styles. The optional
    demangling style argument can be used to choose an
    appropriate demangling style for your compiler.

--no-demangle
    Do not demangle low-level symbol names. This is the default.

--recurse-limit
--no-recurse-limit
--recursion-limit
--no-recursion-limit
    Enables or disables a limit on the amount of recursion
    performed whilst demangling strings. Since the name mangling
    formats allow for an infinite level of recursion it is
    possible to create strings whose decoding will exhaust the
    amount of stack space available on the host machine,
    triggering a memory fault. The limit tries to prevent this
    from happening by restricting recursion to 2048 levels of
    nesting.

    The default is for this limit to be enabled, but disabling it
    may be necessary in order to demangle truly complicated
    names. Note however that if the recursion limit is disabled
    then stack exhaustion is possible and any bug reports about
    such an event will be rejected.

-D
--dynamic
    Display the dynamic symbols rather than the normal symbols.
    This is only meaningful for dynamic objects, such as certain
    types of shared libraries.

-f format
--format=format
    Use the output format format, which can be "bsd", "sysv",
    "posix" or "just-symbols". The default is "bsd". Only the
    first character of format is significant; it can be either
    upper or lower case.

```



```

-g
--extern-only
    Display only external symbols.

-h
--help
    Show a summary of the options to nm and exit.

--ifunc-chars=CHARS
    When display GNU indirect function symbols nm will default to
    using the "i" character for both local indirect functions and
    global indirect functions. The --ifunc-chars option allows
    the user to specify a string containing one or two
    characters. The first character will be used for global
    indirect function symbols and the second character, if
    present, will be used for local indirect function symbols.

j    The same as --format=just-symbols.

-l
--line-numbers
    For each symbol, use debugging information to try to find a
    filename and line number. For a defined symbol, look for the
    line number of the address of the symbol. For an undefined
    symbol, look for the line number of a relocation entry which
    refers to the symbol. If line number information can be
    found, print it after the other symbol information.

--inlines
    When option -l is active, if the address belongs to a
    function that was inlined, then this option causes the source
    information for all enclosing scopes back to the first non-
    inlined function to be printed as well. For example, if
    "main" inlines "calleel" which inlines "callee2", and address
    is from "callee2", the source information for "calleel" and
    "main" will also be printed.

-n
-v
--numeric-sort
    Sort symbols numerically by their addresses, rather than
    alphabetically by their names.

-p
--no-sort
    Do not bother to sort the symbols in any order; print them in
    the order encountered.

-P

```

--portability
 Use the POSIX.2 standard output format instead of the default format. Equivalent to **-f posix**.

-r
--reverse-sort
 Reverse the order of the sort (whether numeric or alphabetic); **let** the last come first.

-S
--print-size
 Print both value and size of defined symbols **for** the "bsd" output style. This option has no effect **for** object formats that **do** not record symbol sizes, unless **--size-sort** is also used **in** which **case** a calculated size is displayed.

-s
--print-arnamap
 When listing symbols from archive members, include the index: a mapping (stored **in** the archive by ar or ranlib) of which modules contain definitions **for** which names.

-t radix
--radix=radix
 Use radix as the radix **for** printing the symbol values. It must be d **for** decimal, o **for** octal, or x **for** hexadecimal.

-u
--undefined-only
 Display only undefined symbols (those external to each object file). By default both defined and undefined symbols are displayed.

-U
--defined-only
 Display only defined symbols **for** each object file. By default both defined and undefined symbols are displayed.

-V
--version
 Show the version number of nm and exit.

-X This option is ignored **for** compatibility with the AIX version of nm. It takes one parameter which must be the string 32_64. The default mode of AIX nm corresponds to **-X 32**, which is not supported by GNU nm.

--plugin name
 Load the plugin called name to add support **for** extra target types. This option is only available **if** the toolchain has

been built with plugin support enabled.

If `--plugin` is not provided, but plugin support has been enabled **then** nm iterates over the files **in** `${libdir}/bfd-plugins` **in** alphabetic order and the first plugin that claims the object **in** question is used.

Please note that this plugin search directory is not the one used by ld's `-plugin` option. In order to make nm use the linker plugin it must be copied into the `${libdir}/bfd-plugins` directory. For GCC based compilations the linker plugin is called `liblto_plugin.so.0.0.0`. For Clang based compilations it is called `LLVMgold.so`. The GCC plugin is always backwards compatible with earlier versions, so it is sufficient to just copy the newest one.

`--size-sort`

Sort symbols by size. For ELF objects symbol sizes are **read** from the ELF, **for** other object types the symbol sizes are computed as the difference between the value of the symbol and the value of the symbol with the next higher value. If the "bsd" output format is used the size of the symbol is printed, rather than the value, and `-S` must be used **in** order both size and value to be printed.

Note - this option does not work **if** `--undefined-only` has been enabled as undefined symbols have no size.

`--special-syms`

Display symbols which have a target-specific special meaning. These symbols are usually used by the target **for** some special processing and are not normally helpful when included **in** the normal symbol lists. For example **for** ARM targets this option would skip the mapping symbols used to mark transitions between ARM code, THUMB code and data.

`--synthetic`

Include synthetic symbols **in** the output. These are special symbols created by the linker **for** various purposes. They are not shown by default since they are not part of the binary's original **source** code.

`--unicode=[default|invalid|locale|escape|hex|highlight]`

Controls the display of UTF-8 encoded multibyte characters **in** strings. The default (`--unicode=default`) is to give them no special treatment. The `--unicode=locale` option displays the sequence **in** the current locale, which may or may not support them. The options `--unicode=hex` and `--unicode=invalid` display them as hex byte sequences enclosed by either angle brackets or curly braces.

The `--unicode=escape` option displays them as escape sequences (`\uxxxx`) and the `--unicode=highlight` option displays them as escape sequences highlighted **in** red (**if** supported by the output device). The colouring is intended to draw attention to the presence of unicode sequences where they might not be expected.

`-W`

`--no-weak`

Do not display weak symbols.

`--with-symbol-versions`

`--without-symbol-versions`

Enables or disables the display of symbol version information. The version string is displayed as a suffix to the symbol name, preceded by an `@` character. For example `foo@VER_1`. If the version is the default version to be used when resolving unversioned references to the symbol **then** it is displayed as a suffix preceded by two `@` characters. For example `foo@@VER_2`. By default, symbol version information is displayed.

`--target=bfdname`

Specify an object code format other than your system's default format.

`@file`

Read **command**-line options from file. The options **read** are inserted **in** place of the original `@file` option. If file does not exist, or cannot be **read**, **then** the option will be treated literally, and not removed.

Options **in** file are separated by whitespace. A whitespace character may be included **in** an option by surrounding the entire option **in** either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional `@file` options; any such options will be processed recursively.

SEE ALSO

`ar(1)`, `objdump(1)`, `ranlib(1)`, and the Info entries **for** `binutils`.

COPYRIGHT

Copyright (c) 1991-2024 Free Software Foundation, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is

included **in** the section entitled "GNU_Free_Documentation License".

COLOPHON

This page is part of the binutils (a collection of tools **for** working with executable binaries) project. Information about the project can be found at <http://www.gnu.org/software/binutils/>. If you have a bug report **for** this manual page, see http://sourceware.org/bugzilla/enter_bug.cgi?product=binutils. This page was obtained from the tarball binutils-2.42.tar.gz fetched from <https://ftp.gnu.org/gnu/binutils/> on 2024-06-14. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

binutils-2.42

2024-06-14

NM(1)

3.8 strace: Trace System Calls and Signals

NAME

strace - trace system calls and signals

SYNOPSIS

```
strace [-ACdffhikkqrrtttTvVwxyyYzZ] [-a column] [-b execve]
      [-e expr]... [-I n] [-o file] [-O overhead] [-p pid]...
      [-P path]... [-s strsize] [-S sortby] [-U columns]
      [-X format] [--seccomp-bpf]
      [--stack-trace-frame-limit=limit] [--syscall-limit=limit]
      [--secontext[=format]] [--tips[=format]] { -p pid | [-DDD]
      [-E var[=val]]... [-u username] command [args] }
```

```
strace -c [-dfwzZ] [-b execve] [-e expr]... [-I n] [-O overhead]
      [-p pid]... [-P path]... [-S sortby] [-U columns]
      [--seccomp-bpf] [--syscall-limit=limit] [--tips[=format]]
      { -p pid | [-DDD] [-E var[=val]]... [-u username] command
      [args] }
```

```
strace --tips[=format]
```

DESCRIPTION

In the simplest **case** strace runs the specified **command** until it exits. It intercepts and records the system calls which are called by a process and the signals which are received by a process. The name of each system call, its arguments and its **return** value are printed on standard error or to the file specified with the -o option.

strace is a useful diagnostic, instructional, and debugging tool. System administrators, diagnosticians and trouble-shooters will find it invaluable **for** solving problems with programs **for** which

the **source** is not readily available since they **do** not need to be recompiled **in** order to trace them. Students, hackers and the overly-curious will find that a great deal can be learned about a system and its system calls by tracing even ordinary programs. And programmers will find that since system calls and signals are events that happen at the user/kernel interface, a close examination of this boundary is very useful **for** bug isolation, sanity checking and attempting to capture race conditions.

Each line **in** the trace contains the system call name, followed by its arguments **in** parentheses and its **return** value. An example from stracing the **command** "cat_/dev/null" is:

```
open("/dev/null", O_RDONLY) = 3
```

Errors (typically a **return** value of -1) have the **errno** symbol and error string appended.

```
open("/foo/bar", O_RDONLY) = -1 ENOENT (No such file or directory)
```

Signals are printed as signal symbol and decoded siginfo structure. An excerpt from stracing and interrupting the **command** "sleep_666" is:

```
sigsuspend([] <unfinished ...>
--- SIGINT {si_signo=SIGINT, si_code=SI_USER, si_pid=...} ---
+++ killed by SIGINT +++
```

If a system call is being executed and meanwhile another one is being called from a different thread/process **then** strace will try to preserve the order of those events and mark the ongoing call as being unfinished. When the call returns it will be marked as resumed.

```
[pid 28772] select(4, [3], NULL, NULL, NULL <unfinished ...>
[pid 28779] clock_gettime(CLOCK_REALTIME, {tv_sec=1130322148,
tv_nsec=39770000}) = 0
[pid 28772] <... select resumed> )      = 1 (in [3])
```

Interruption of a (restartable) system call by a signal delivery is processed differently as kernel terminates the system call and also arranges its immediate reexecution after the signal handler completes.

```
read(0, 0x7ffff72cf5cf, 1)      = ? ERESTARTSYS (To be
    restarted)
--- SIGALRM {si_signo=SIGALRM, si_code=SI_KERNEL} ---
rt_sigreturn({mask=[]})        = 0
read(0, "", 1)                  = 0
```

Arguments are printed **in** symbolic form with passion. This example shows the shell performing ">>xyzyz" output redirection:

```
open("xyzyz", O_WRONLY|O_APPEND|O_CREAT, 0666) = 3
```

Here, the second and the third argument of `open(2)` are decoded by breaking down the flag argument into its three bitwise-OR constituents and printing the mode value **in** octal by tradition. Where the traditional or native usage differs from ANSI or POSIX, the latter forms are preferred. In some cases, `strace` output is proven to be more readable than the source.

Structure pointers are dereferenced and the members are displayed as appropriate. In most cases, arguments are formatted **in** the most C-like fashion possible. For example, the essence of the **command** "`ls -l /dev/null`" is captured as:

```
lstat("/dev/null", {st_mode=S_IFCHR|0666, st_rdev=makedev(0x1, 0x3), ...}) = 0
```

Notice how the '`struct stat`' argument is dereferenced and how each member is displayed symbolically. In particular, observe how the `st_mode` member is carefully decoded into a bitwise-OR of symbolic and numeric values. Also notice **in** this example that the first argument to `lstat(2)` is an input to the system call and the second argument is an output. Since output arguments are not modified **if** the system call fails, arguments may not always be dereferenced. For example, retrying the "`ls -l`" example with a non-existent file produces the following line:

```
lstat("/foo/bar", 0xb004) = -1 ENOENT (No such file or directory)
```

In this **case** the porch light is on but nobody is home.

Syscalls unknown to `strace` are printed raw, with the unknown system call number printed **in** hexadecimal form and prefixed with "`syscall_`":

```
syscall_0xbad(0x1, 0x2, 0x3, 0x4, 0x5, 0x6) = -1 ENOSYS (Function not implemented)
```

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(3)` library routine is reading the password file:

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

While structures are annotated using curly braces, pointers to basic types and arrays are printed using square brackets with commas separating the elements. Here is an example from the **command** `id(1)` 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

General

-e expr

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

```
[qualifier=][!]value[,value]...
```

where **qualifier** is one of **trace** (or **t**), **trace-fds** (or **trace-fd** or **fd** or **fds**), **abbrev** (or **a**), **verbose** (or **v**), **raw** (or **x**), **signal** (or **signals** or **s**), **read** (or **reads** or **r**), **write** (or **writes** or **w**), **fault**, **inject**, **status**, **quiet** (or **silent** or **silence** or **q**), **secontext**, **decode-fds** (or **decode-fd**), **decode-pids** (or **decode-pid**), or **kvm**, 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, **-e open** means literally **-e trace=open** which **in** turn means trace only the open system call. By contrast, **-e trace=!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.

Startup


```

-E var=val
--env=var=val
    Run command with var=val in its list of environment
    variables.

-E var
--env=var
    Remove var from the inherited list of environment
    variables before passing it on to the command.

-p pid
--attach=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 many processes in addition to command
    (which is optional if at least one -p option is given).
    Multiple process IDs, separated by either comma (','),
    space(" "), tab, or newline character, can be provided as
    an argument to a single -p option, so, for example, -p
    "$(pidof PROG)" and -p "$(pgrep PROG)" syntaxes are
    supported.

-----u_username
----- --user=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.
-----u_UID:GID
----- --user=UID:GID
    Alternative syntax where the program is started with
    exactly the given user and group IDs, and an empty list of
    supplementary groups. In this case, user and group name
    lookups are not performed.

----- --argv0=name
    Set argv[0] of the command being executed to name. Useful
    for tracing multi-call executables which interpret
    argv[0], such as busybox or kmod.

Tracing
-----b_syscall
----- --detach-on=syscall
    If specified syscall is reached, detach from traced
    process. Currently, only execve(2) syscall is supported.

```

```

This_option_is_useful_if_you_want_to_trace_multi-threaded
process_and_therefore_require_-f,_but_don't_want_to_trace
its_(potentially_very_complex)_children.

-D
--daemonize
--daemonize=grandchild
Run_tracer_process_as_a_grandchild,_not_as_the_parent_of
the_tracee. This_reduces_the_visible_effect_of_strace_by
keeping_the_tracee_a_direct_child_of_the_calling_process.

-DD
--daemonize=pgroup
--daemonize=pgrp
Run_tracer_process_as_tracee's_grandchild_in_a_separate
process_group. In_addition_to_reduction_of_the_visible
effect_of_strace,_it_also_avoids_killing_of_strace_with
kill(2)_issued_to_the_whole_process_group.

-DDD
--daemonize=session
Run_tracer_process_as_tracee's_grandchild_in_a_separate
session("true daemonisation"). In_addition_to_reduction
of_the_visible_effect_of_strace,_it_also_avoids_killing_of
strace_upon_session_termination.

-f
--follow-forks
Trace_child_processes_as_they_are_created_by_currently
traced_processes_as_a_result_of_the_fork(2),_vfork(2)_and
clone(2)_system_calls. Note_that_-p_PID_-f_will_attach
all_threads_of_process_PID_if_it_is_multi-threaded,_not
only_thread_with_thread_id=_PID.

--output-separately
If_the_--output=filename_option_is_in_effect,_each
processes_trace_is_written_to_filename.pid_where_pid_is
the_numeric_process_id_of_each_process.

-ff
--follow-forks_--output-separately
Combine_the_effects_of_--follow-forks_and
--output-separately_options. This_is_incompatible_with
-c,_since_no_per-process_counts_are_kept.

One_might_want_to_consider_using_strace-log-merge(1)_to
obtain_a_combined_strace_log_view.

-I_interruptible
--interruptible=interruptible

```

```

When strace can be interrupted by signals (such as
pressing CTRL-C).

1, anywhere
no signals are blocked;
2, waiting
fatal signals are blocked while decoding syscall
(default);
3, never
fatal signals are always blocked (default if -o
FILE_PROG);
4, never_tstp
fatal signals and SIGTSTP (CTRL-Z) are always
blocked (useful to make strace -o FILE_PROG not
stop on CTRL-Z, default if -D).

--syscall-limit=limit
Detach all tracees when limit number of syscalls have been
captured. Syscalls filtered out via --trace, --trace-path
or --status options are not considered when keeping track
of the number of syscalls that are captured.

--kill-on-exit
Apply PT_TRACE_0_EXITKILL ptrace option to all tracee
processes (which sends a SIGKILL signal to the tracee if
the tracer exits) and do not detach them on cleanup so
they will not be left running after the tracer exit.
--kill-on-exit is not compatible with -p/--attach options.

Filtering
-e,trace=syscall_set
-e,t=syscall_set
--trace=syscall_set
Trace only the specified set of system calls. syscall_set
is defined as [!]value[,value], and value can be one of
the following:

syscall
Trace specific syscall, specified by its name (see
syscalls(2) for a reference, but also see NOTES).

?value
Question mark before the syscall qualification
allows suppression of error in case no syscalls
matched the qualification provided.

value@64
Limit the syscall specification described by value
to 64-bit personality.

value@32

```

```

Limit the syscall specification described by value
to 32-bit personality.

value@x32
Limit the syscall specification described by value
to x32 personality.

all Trace all system calls.

/regex Trace only those system calls that match the regex.
You can use POSIX Extended Regular Expression
syntax (see regex(7)).

%file
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(2) in the list. Betchya
woulda forgot that one. The syntax without a
preceding percent sign ("-e trace=file") is
deprecated.

%process
process
Trace system calls associated with process
lifecycle (creation, exec, termination). The
syntax without a preceding percent sign ("-e
trace=process") is deprecated.

%net
%network
network
Trace all the network related system calls. The
syntax without a preceding percent sign ("-e
trace=network") is deprecated.

%signal
signal Trace all signal related system calls. The syntax
without a preceding percent sign ("-e
trace=signal") is deprecated.

%ipc
ipc Trace all IPC related system calls. The syntax
without a preceding percent sign ("-e trace=ipc")
is deprecated.

%desc

```

```

desc Trace_all_file_descriptor_related_system_calls.
The_syntax_without_a_preceding_percent_sign("-e
trace=desc")_is_deprecated.

%memory
memory Trace_all_memory_mapping_related_system_calls. The
syntax_without_a_preceding_percent_sign("-e
trace=memory")_is_deprecated.

%creds Trace_system_calls_that_read_or_modify_user_and
group_identifiers_or_capability_sets.

%stat Trace_stat_syscall_variants.

%lstat Trace_lstat_syscall_variants.

%fstat Trace_fstat,_fstatat,_and_statx_syscall_variants.

%%stat Trace_syscalls_used_for_requesting_file_status
(stat,_lstat,_fstat,_fstatat,_statx,_and_their
variants).

%statfs
Trace_statfs,_statfs64,_statvfs,_osf_statfs,_and
osf_statfs64_system_calls. The_same_effect_can_be
achieved_with_e_trace=/^(.*)?statv?fs_regular
expression.

%fstatfs
Trace_fstatfs,_fstatfs64,_fstatvfs,_osf_fstatfs,
and_osf_fstatfs64_system_calls. The_same_effect
can_be_achieved_with_e_trace=/fstatv?fs_regular
expression.

%%statfs
Trace_syscalls_related_to_file_system_statistics
(statfs-like,_fstatfs-like,_and_ustat). The_same
effect_can_be_achieved_with
-e_trace=/statv?fs|fsstat|ustat_regular_expression.

%clock Trace_system_calls_that_read_or_modify_system
clocks.

%pure Trace_syscalls_that_always_succeed_and_have_no
arguments. Currently,_this_list_includes
arc_gettls(2),_getdtablesize(2),_getegid(2),
getegid32(2),_geteuid(2),_geteuid32(2),_getgid(2),
getgid32(2),_getpagesize(2),_getpgrp(2),_getpid(2),
getppid(2),_get_thread_area(2)_on_architectures
other_than_x86),_gettid(2),_get_tls(2),_getuid(2),

```

```

getuid(2),_getxgid(2),_getxpid(2),_getxuid(2),
kern_features(2),_and_metag_get_tls(2)_syscalls.

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-fd=set
-e_trace-fds=set
-e_fd=set
-e_fds=set
--trace-fds=set
Trace_only_the_syscalls_that_operate_on_the_specified
subset_of_(non-negative)_file_descriptors._Note_that
usage_of_this_option_also_filters_out_all_the_syscalls
that_do_not_operate_on_file_descriptors_at_all._Applies
in_(inclusive)_disjunction_with_the_--trace-path_option.

-e_signal=set
-e_signals=set
-e_s=set
--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_status=set
--status=set
Print_only_system_calls_with_the_specified_return_status.
The_default_is_status=all._When_using_the_status
qualifier,_because_strace_waits_for_system_calls_to_return
before_deciding_whether_they_should_be_printed_or_not,_the
traditional_order_of_events_may_not_be_preserved_anymore.
If_two_system_calls_are_executed_by_concurrent_threads,
strace_will_first_print_both_the_entry_and_exit_of_the
first_system_call_to_exit,_regardless_of_their_respective
entry_time._The_entry_and_exit_of_the_second_system_call
to_exit_will_be_printed_afterwards._Here_is_an_example
when_select(2)_is_called,_but_a_different_thread_calls
clock_gettime(2)_before_select(2)_finishes:

[pid_28779]_1130322148.939977_clock_gettime(CLOCK_REALTIME,_
{1130322148,_939977000})_=0
[pid_28772]_1130322148.438139_select(4,_[3],_NULL,_NULL,_
NULL)_=_1_(in_[3])

set_can_include_the_following_elements:

```

```

successful
Trace system calls that returned without an error
code. The -z option has the effect of
status=successful.
failed Trace system calls that returned with an error
code. The -Z option has the effect of
status=failed.
unfinished
Trace system calls that did not return. This might
happen, for example, due to an execve call in a
neighbour thread.
unavailable
Trace system calls that returned but strace failed
to fetch the error status.
detached
Trace system calls for which strace detached before
the return.

-P path
--trace-path=path
Trace only system calls accessing path. Multiple -P
options can be used to specify several paths. Applies in
(inclusive) disjunction with the --trace-fds option.

-Z
--successful-only
Print only syscalls that returned without an error code.

-Z
--failed-only
Print only syscalls that returned with an error code.

Output format
-a column
--columns=column
Align return values in a specific column (default column
40).

-e abbrev=syscall_set
-e a=syscall_set
--abbrev=syscall_set
Abbreviate the output from printing each member of large
structures. The syntax of the syscall_set specification
is the same as in the -e trace option. The default is
abbrev=all. The -v option has the effect of abbrev=None.

-e verbose=syscall_set
-e v=syscall_set
--verbose=syscall_set

```

```

Dereference structures for the specified set of system
calls. The syntax of the syscall set specification is the
same as in the -e trace option. The default is
verbose=all.

-e raw=syscall_set
-e x=syscall_set
-- raw=syscall_set
Print raw, undecoded arguments for the specified set of
system calls. The syntax of the syscall set specification
is the same as in the -e trace option. 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. See also -X raw option.

-e read=set
-e reads=set
-e r=set
-- 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
-e writes=set
-e w=set
-- 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.

-e quiet=set
-e silent=set
-e silence=set
-e q=set
-- quiet=set
-- silent=set
-- silence=set
Suppress various information messages. The default is
quiet=none. set can include the following elements:

attach Suppress messages about attaching and detaching ("[
Process NNNN attached ]", "[ Process NNNN detached

```



```

]").
exit Suppress_messages_about_process_exits("+++ exited
with SSS +++").
path-resolution
Suppress_messages_about_resolution_of_paths
provided_via_the_P_option("Requested path ..."
resolved into "...").
personality
Suppress_messages_about_process_personality_changes
("[ Process PID=NNNN runs in PPP mode. ]").
thread-execve
superseded
Suppress_messages_about_process_being_superseded_by
execve(2)_in_another_thread("+++ superseded by
execve in pid NNNN +++").

-e_decode-fds=set
--decode-fds=set
Decode_various_information_associated_with_file
descriptors. The_default_is_decode-fds=none. set_can
include_the_following_elements:

path Print_file_paths. Also_enables_printing_of
tracee's_current_working_directory_when_AT_FDCWD
constant_is_used.
socket Print_socket_protocol-specific_information,
dev Print_character/block_device_numbers.
pidfd Print_PIDs_associated_with_pidfd_file
descriptors.
signalfd Print_signal_masks_associated_with_signalfd_file
descriptors.

-e_decode-pids=set
--decode-pids=set
Decode_various_information_associated_with_process_IDs
(and_also_thread_IDs,_process_group_IDs,_and_session_IDs).
The_default_is_decode-pids=none. set_can_include_the
following_elements:

comm Print_command_names_associated_with_thread_or
process_IDs.
pidns Print_thread,_process,_process_group,_and_session
IDs_in_strace's_PID_namespace_if_the_tracee_is_in
a_different_PID_namespace.

-e_kvm=vcpu
--kvm=vcpu
Print_the_exit_reason_of_kvm_vcpu. Requires_Linux_kernel
version_4.16.0_or_higher.

```

```

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

- n
--syscall-number
Print the syscall number.

- k
--stack-trace[=symbol]
Print the execution stack trace of the traced processes
after each system call.

- kk
--stack-trace=source
Print the execution stack trace and source code
information of the traced processes after each system
call. This option expects the target program is compiled
with appropriate debug options: "-g"(gcc), or "-g
-gdwarf-aranges"(clang).

--stack-trace-frame-limit=limit
Print no more than this amount of stack trace frames when
backtracing a system call (the default is 256). Use this
option with the --stack-trace (or -k) option.

-o filename
--output=filename
Write the trace output to the file filename rather than to
stderr. filename.pid form is used if -ff option is
supplied. If the argument begins with '|' or '!', 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. The latter is not
compatible with -ff option currently.

-A
--output-append-mode
Open the file provided in the -o option in append mode.

-q
--quiet
--quiet=attach, personality
Suppress messages about attaching, detaching, and
personality changes. This happens automatically when
output is redirected to a file and the command is run
directly instead of attaching.

```

```

-qq
--quiet=attach, personality, exit
Suppress messages attaching, detaching, personality
changes, and about process exit status.

-qqq
--quiet=all
Suppress all suppressible messages (please refer to the -e
quiet option description for the full list of suppressible
messages).

-r
--relative-timestamps[=precision]
Print a relative timestamp upon entry to each system call.
This records the time difference between the beginning of
successive system calls. precision can be one of s (for
seconds), ms (milliseconds), us (microseconds), or ns
(nanoseconds), and allows setting the precision of time
value being printed. Default is us (microseconds). Note
that since -r option uses the monotonic clock time for
measuring time difference and not the wall clock time, its
measurements can differ from the difference in time
reported by the -t option.

-s, strsize
--string-limit=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.

--absolute-timestamps=[[format:]format], [[precision:]precision]]
--timestamps=[[format:]format], [[precision:]precision]]
Prefix each line of the trace with the wall clock time in
the specified format with the specified precision. format
can be one of the following:

none No time stamp is printed. Can be used to override
the previous setting.
time Wall clock time (strftime(3) format string is %T).
unix Number of seconds since the epoch (strftime(3)
format string is %s).

precision can be one of s (for seconds), ms
(milliseconds), us (microseconds), or ns (nanoseconds).
Default arguments for the option are
format:time, precision:s.

-t
--absolute-timestamps
Prefix each line of the trace with the wall clock time.

```

```

-tt
-absolute-timestamps=precision:us
If_given_twice,_the_time_printed_will_include_the
microseconds.

-ttt
-absolute-timestamps=format:unix,precision:us
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
-syscall-times[=precision]
Show_the_time_spent_in_system_calls._This_records_the
time_difference_between_the_beginning_and_the_end_of_each
system_call._precision_can_be_one_of_s_(for_seconds),_ms
(milliseconds),_us_(microseconds),_or_ns_(nanoseconds),
and_allows_setting_the_precision_of_time_value_being
printed._Default_is_us_(microseconds).

-V
-no-abbrev
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.

--strings-in-hex[=option]
Control_usage_of_escape_sequences_with_hexadecimal_numbers
in_the_printed_strings._Normally_(when_no
--strings-in-hex_or_-x_option_is_supplied),_escape
sequences_are_used_to_print_non-printable_and_non-ASCII
characters_(that_is,_characters_with_a_character_code_less
than_32_or_greater_than_127),_or_to_disambiguate_the
output_(so,_for_quotes_and_other_characters_that_encase
the_printed_string,_for_example,_angle_brackets,_in_case
of_file_descriptor_path_output);_for_the_former_use_case,
unless_it_is_a_white_space_character_that_has_a_symbolic
escape_sequence_defined_in_the_C_standard_(that_is,_
"\textbackslash t"
for_a_horizontal_tab,_"\textbackslash n"_for_a_newline,_"
"\textbackslash v"_for_a
vertical_tab,_"\textbackslash f"_for_a_form_feed_page_break,_and_
"\textbackslash r"
for_a_carriage_return)_are_printed_using_escape_sequences
with_numbers_that_correspond_to_their_byte_values,_with
octal_number_format_being_the_default._option_can_be_one
of_the_following:

```

```

        none      Hexadecimal numbers are not used in the output at
        all.      When there is a need to emit an escape
        sequence, octal numbers are used.
        non-ascii-chars
        Hexadecimal numbers are used instead of octal in
        the escape sequences.
        non-ascii
        Strings that contain non-ASCII characters are
        printed using escape sequences with hexadecimal
        numbers.
        all       All strings are printed using escape sequences with
        hexadecimal numbers.

        When the option is supplied without an argument, all is
        assumed.

        -X
        --strings-in-hex=non-ascii
        Print all non-ASCII strings in hexadecimal string format.

        -XX
        --strings-in-hex[=all]
        Print all strings in hexadecimal string format.

        -X_format
        --const-print-style=format
        Set the format for printing of named constants and flags.
        Supported format values are:

        raw       Raw number output, without decoding.
        abbrev    Output a named constant or a set of flags instead
        of the raw number if they are found. This is the
        default strace behaviour.
        verbose
        Output both the raw value and the decoded string
        (as a comment).

        -y
        --decode-fds
        --decode-fds=path
        Print paths associated with file descriptor arguments and
        with the AT_FDCWD constant.

        -yy
        --decode-fds=all
        Print all available information associated with file
        descriptors: protocol-specific information associated with
        socket file descriptors, block/character device number
        associated with device file descriptors, and PIDs

```

```

associated_with_pidfd_file_descriptors.

-----pidns-translation
-----decode-pids=pidns
-----If_strace_and_tracee_are_in_different_PID_namespaces,
-----print_PIDs_in_strace's_namespace,_too.

-----Y
-----decode-pids=comm
-----Print_command_names_for_PIDs.

-----secontext[=format]
-----e_secontext=format
-----When_SELinux_is_available_and_is_not_disabled,_print_in
-----square_brackets_SELinux_contexts_of_processes,_files,_and
-----descriptors._The_format_argument_is_a_comma-separated
-----list_of_items_being_one_of_the_following:

-----full-----Print_the_full_context_(user,_role,_type
-----level_and_category).
-----mismatch-----Also_print_the_context_recorded_by_the
-----SELinux_database_in_case_the_current
-----context_differs._The_latter_is_printed
-----after_two_exclamation_marks_(!!).

-----The_default_value_for_--secontext_is_!full,mismatch_which
-----prints_only_the_type_instead_of_full_context_and_doesn't
-----check_for_context_mismatches.

-----always-show-pid
-----Show_PID_prefix_also_for_the_process_started_by_strace.
-----Implied_when_-f_and_-o_are_both_specified.

Statistics
-----C
-----summary-only
-----Count_time,_calls,_and_errors_for_each_system_call_and
-----report_a_summary_on_program_exit,_suppressing_the_regular
-----output._This_attempts_to_show_system_time_(CPU_time_spent
-----running_in_the_kernel)_independent_of_wall_clock_time._If
-----c_is_used_with_-f,_only_aggregate_totals_for_all_traced
-----processes_are_kept.

-----C
-----summary
-----Like_-c_but_also_print_regular_output_while_processes_are
-----running.

-----O_overhead
-----summary-syscall-overhead=overhead

```

```

Set the overhead for tracing system calls to overhead.
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.

The format of overhead specification is described in
section Time specification format description.

-S, sortby
--summary-sort-by=sortby
Sort the output of the histogram printed by the -c option
by the specified criterion. Legal values are time (or
time-percent or time-total or total-time), min-time (or
shortest or time-min), max-time (or longest or time-max),
avg-time (or time-avg), calls (or count), errors (or
error), name (or syscall or syscall-name), and nothing (or
none); default is time.

-U, columns
--summary-columns=columns
Configure a set (and order) of columns being shown in the
call summary. The columns argument is a comma-separated
list with items being one of the following:

time-percent (or time)
Percentage of cumulative time consumed by a
specific system call.
total-time (or time-total)
Total system (or wall clock, if -w option is
provided) time consumed by a specific system call.
min-time (or shortest or time-min)
Minimum observed call duration.
max-time (or longest or time-max)
Maximum observed call duration.
avg-time (or time-avg)
Average call duration.
calls (or count)
Call count.
errors (or error)
Error count.
name (or syscall or syscall-name)
Syscall name.

The default value is
time-percent, total-time, avg-time, calls, errors, name. If
the name field is not supplied explicitly, it is added as

```

```

the_last_column.

-W
--summary-wall-clock
Summarise the time difference between the beginning and
end of each system call. The default is to summarise the
system time.

Tampering
-e_inject=syscall_set[:error=errno|:retval=value][:signal=sig]
[:syscall=syscall][:delay_enter=delay][:delay_exit=delay]
[:poke_enter=@argN=DATAN,@argM=DATAM...]
[:poke_exit=@argN=DATAN,@argM=DATAM...][:when=expr]
--inject=syscall_set[:error=errno|:retval=value][:signal=sig]
[:syscall=syscall][:delay_enter=delay][:delay_exit=delay]
[:poke_enter=@argN=DATAN,@argM=DATAM...]
[:poke_exit=@argN=DATAN,@argM=DATAM...][:when=expr]
Perform syscall tampering for the specified set of
syscalls. The syntax of the syscall set specification is
the same as in the -e_trace option.

At least one of error, retval, signal, delay_enter,
delay_exit, poke_enter, or poke_exit options has to be
specified. error and retval are mutually exclusive.

If :error=errno option is specified, a fault is injected
into a syscall invocation: the syscall number is replaced
by -1 which corresponds to an invalid syscall (unless a
syscall is specified with :syscall=option), and the error
code is specified using a symbolic errno value like ENOSYS
or a numeric value within 1..4095 range.

If :retval=value option is specified, success injection is
performed: the syscall number is replaced by -1, but a
bogus success value is returned to the callee.

If :signal=sig option is specified with either a symbolic
value like SIGSEGV or a numeric value within 1..SIGRTMAX
range, that signal is delivered on entering every syscall
specified by the set.

If :delay_enter=delay or :delay_exit=delay options are
specified, delay injection is performed: the tracee is
delayed by time period specified by delay on entering or
exiting the syscall, respectively. The format of delay
specification is described in section Time specification
format description.

If :poke_enter=@argN=DATAN,@argM=DATAM... or
:poke_exit=@argN=DATAN,@argM=DATAM... options are

```



```

specified, tracee's memory at locations, pointed to by
system_call_arguments_argN_and_argM (going from arg1 to
arg7) is overwritten by data_DATAN and DATAM (specified in
hexadecimal format; for example
:poke_enter=@arg1=0000DEAD0000BEEF). :poke_enter modifies
memory on syscall_enter, and :poke_exit on_exit.

If :signal=sig_option is specified without :error=errno,
:retval=value or :delay_{enter,exit}=usecs_options, then
only a signal sig is delivered without a syscall_fault or
delay injection. Conversely, :error=errno or
:retval=value option without :delay_enter=delay,
:delay_exit=delay or :signal=sig_options injects a fault
without delivering a signal or injecting a delay, etc.

If :signal=sig_option is specified together with
:error=errno or :retval=value, then both injection of a
fault or success and signal delivery are performed.

if :syscall=syscall_option is specified, the corresponding
syscall with no side effects is injected instead of -1.
Currently, only "pure" (see -e_trace=%pure_description)
syscalls can be specified there.

Unless a :when=expr subexpression is specified, an
injection is being made into every invocation of each
syscall from the set.

The format of the subexpression is:

first[..last][+[step]]

Number_first stands for the first invocation number in the
range, number_last stands for the last invocation number
in the range, and step stands for the step between two
consecutive invocations. The following combinations are
useful:

first For every syscall from the set, perform an
injection for the syscall invocation number first
only.
first..last For every syscall from the set, perform an
injection for the syscall invocation number first
and all subsequent invocations until the invocation
number last (inclusive).
first+ For every syscall from the set, perform injections
for the syscall invocation number first and all
subsequent invocations.
first..last+

```

```

For_every_syscall_from_the_set,_perform_injections
for_the_syscall_invocation_number_first_and_all
subsequent_invocations_until_the_invocation_number
last_(inclusive).
first+step
For_every_syscall_from_the_set,_perform_injections
for_syscall_invocations_number_first,_first+step,
first+step+step,_and_so_on.
first..last+step
Same_as_the_previous,_but_consider_only_syscall
invocations_with_numbers_up_to_last_(inclusive).

For_example,_to_fail_each_third_and_subsequent_chdir
syscalls_with_ENOENT,_use
-e_inject=chdir:error=ENOENT:when=3+.

The_valid_range_for_numbers_first_and_step_is_1..65535,
and_for_number_last_is_1..65534.

An_injection_expression_can_contain_only_one_error=_or
retval=_specification,_and_only_one_signal=_specification.
If_an_injection_expression_contains_multiple_when=
specifications,_the_last_one_takes_precedence.

Accounting_of_syscalls_that_are_subject_to_injection_is
done_per_syscall_and_per_tracee.

Specification_of_syscall_injection_can_be_combined_with
other_syscall_filtering_options,_for_example,_-P
/dev/urandom_-e_inject=file:error=ENOENT.

-e_fault=syscall_set[:error=errno][:when=expr]
--fault=syscall_set[:error=errno][:when=expr]
Perform_syscall_fault_injection_for_the_specified_set_of
syscalls.

This_is_equivalent_to_more_generic_-e_inject=_expression
with_default_value_of_errno_option_set_to_ENOSYS.

Miscellaneous
-d
--debug
Show_some_debugging_output_of_strace_itself_on_the
standard_error.

-F This_option_is_deprecated._It_is_retained_for_backward
compatibility_only_and_may_be_removed_in_future_releases.
Usage_of_multiple_instances_of_-F_option_is_still
equivalent_to_a_single_-f,_and_it_is_ignored_at_all_if
used_along_with_one_or_more_instances_of_-f_option.

```

```

-h
--help Print the help summary.

--seccomp-bpf
    Try to enable use of seccomp-bpf (see seccomp(2)) to have
    ptrace(2)-stops only when system calls that are being
    traced occur in the traced processes.

    This option has no effect unless -f/--follow-forks is also
    specified. --seccomp-bpf is not compatible with
    --syscall-limit and -b/--detach-on options. It is also
    not applicable to processes attached using -p/--attach
    option.

    An attempt to enable system calls filtering using seccomp-
    bpf may fail for various reasons, e.g. there are too many
    system calls to filter, the seccomp API is not available,
    or strace itself is being traced. In cases when seccomp-
    bpf filter setup failed, strace proceeds as usual and
    stops traced processes on every system call.

    When --seccomp-bpf is activated and -p/--attach option is
    not used, --kill-on-exit option is activated as well.

    Note that in cases when the tracee has another seccomp
    filter that returns an action value with a precedence
    greater than SECCOMP_RET_TRACE, strace --seccomp-bpf will
    not be notified. That is, if another seccomp filter, for
    example, disables the syscall or kills the tracee, then
    strace --seccomp-bpf will not be aware of that syscall
    invocation at all.

--tips[=[id:id],[format:]format]]
    Show strace tips, tricks, and tweaks before exit. id can
    be a non-negative integer number, which enables printing
    of specific tip, trick, or tweak (these ID are not
    guaranteed to be stable), or random (the default), in
    which case a random tip is printed. format can be one of
    the following:

    none      No tip is printed. Can be used to override the
    previous setting.
    compact   Print the tip just big enough to contain all the
    text.
    full      Print the tip in its full glory.

    Default is id:random, format:compact.

-V

```

```

----- --version
----- Print the version number of strace. Multiple instances of
----- the option beyond specific threshold tend to increase
----- Strauss awareness.

Time specification format description
----- Time values can be specified as a decimal floating point number
----- (in a format accepted by strtod(3)), optionally followed by one
----- of the following suffices that specify the unit of time: s
----- (seconds), ms (milliseconds), us (microseconds), or ns
----- (nanoseconds). If no suffix is specified, the value is
----- interpreted as microseconds.

----- The described format is used for -0, -e inject=delay_enter, and
----- -e inject=delay_exit options.
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. Note that parent-
----- child relationship (signal stop notifications, getppid(2) value,
----- etc) between traced process and its parent are not preserved
----- unless -D is used.

----- When using -p without a command, the exit status of strace is
----- zero unless no processes has been attached or 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 regular
----- non-setuid version of strace for ordinary users to use.
MULTIPLE PERSONALITIES SUPPORT
----- On some architectures, strace supports decoding of syscalls for
----- processes that use different ABI rather than the one strace uses.
----- Specifically, in addition to decoding native ABI, strace can
----- decode the following ABIs on the following architectures:

----- [1] When strace is built as an x86_64 application
----- [2] When strace is built as an x32 application
----- [3] Big endian only

```

_____This support is optional and relies on ability to generate and
_____parse structure definitions during the build time. Please refer
_____to the output of the strace -V command in order to figure out
_____what support is available in your strace build ("non-native"
_____refers to an ABI that differs from the ABI strace has):

_____m32-mpers
_____strace can trace and properly decode non-native 32-bit
_____binaries.

_____no-m32-mpers
_____strace can trace, but cannot properly decode non-native
_____32-bit binaries.

_____mx32-mpers
_____strace can trace and properly decode non-native
_____32-on-64-bit binaries.

_____no-mx32-mpers
_____strace can trace, but cannot properly decode non-native
_____32-on-64-bit binaries.

_____If the output contains neither m32-mpers nor no-m32-mpers, then
_____decoding of non-native 32-bit binaries is not implemented at all
_____or not applicable.

_____Likewise, if the output contains neither mx32-mpers nor no-
_____mx32-mpers, then decoding of non-native 32-on-64-bit binaries is
_____not implemented at all or not applicable.

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, the faccessat(2)
_____system call does not have flags argument, and the setrlimit(2)
_____library function uses prlimit64(2) system call on modern
_____ (2.6.38+) kernels. These discrepancies are normal but
_____idiosyncratic characteristics of the system call interface and
_____are accounted for by C library wrapper functions.

_____Some system calls have different names in different architectures
_____and personalities. In these cases, system call filtering and
_____printing uses the names that match corresponding NR_* kernel
_____macros of the tracee's architecture and personality. There are
_____two exceptions from this general rule: arm fadvise64_64(2) ARM
_____syscall and xtensa fadvise64_64(2) Xtensa syscall are filtered

```

and_printed_as_fadvise64_64(2).

On_x32,_syscalls_that_are_intended_to_be_used_by_64-bit_processes
and_not_x32_ones_(for_example,_readv(2),_that_has_syscall_number
19_on_x86_64,_with_its_x32_counterpart_has_syscall_number_515),
but_called_with___X32_SYSCALL_BIT_flag_being_set,_are_designated
with_#64_suffix.

On_some_platforms_a_process_that_is_attached_to_with_the_-p
option_may_observe_a_spurious_EINTR_return_from_the_current
system_call_that_is_not_restartable.__(Ideally,_all_system_calls
should_be_restarted_on_strace_attach,_making_the_attach_invisible
to_the_traced_process,_but_a_few_system_calls_aren't.__(Arguably,
every_instance_of_such_behavior_is_a_kernel_bug.))_This_may_have
an_unpredictable_effect_on_the_process_if_the_process_takes_no
action_to_restart_the_system_call.

As_strace_executes_the_specified_command_directly_and_does_not
employ_a_shell_for_that,_scripts_without_shebang_that_usually_run
just_fine_when_invoked_by_shell_fail_to_execute_with_ENOEXEC
error.__(It_is_advisable_to_manually_supply_a_shell_as_a_command
with_the_script_as_its_argument.
BUGS

Programs_that_use_the_setuid_bit_do_not_have_effective_user_ID
privileges_while_being_traced.

A_traced_process_runs_slowly_(but_check_out_the_--seccomp-bpf
option).

Unless_--kill-on-exit_option_is_used_(or_--seccomp-bpf_option_is
used_in_a_way_that_implies_--kill-on-exit),_traced_processes
which_are_descended_from_command_may_be_left_running_after_an
interrupt_signal_(CTRL-C).

By_using_CLONE_UNTRACED_flag_of_clone_system_call_a_tracee_can
break_the_guarantee_that_--seccomp-bpf_will_not_leave_any
processes_with_a_seccomp_program_installed_for_syscall_filtering
purposes.
HISTORY

The_original_strace_was_written_by_Paul_Kranenburg_for_SunOS_and
was_inspired_by_its_trace_utility.__(The_SunOS_version_of_strace
was_ported_to_Linux_and_enhanced_by_Branko_Lankester,_who_also
wrote_the_Linux_kernel_support.__(Even_though_Paul_released_strace
2.5_in_1992,_Branko's_work_was_based_on_Paul's_strace_1.5_release
from_1991.__(In_1993,_Rick_Sladkey_merged_strace_2.5_for_SunOS_and
the_second_release_of_strace_for_Linux,_added_many_of_the
features_of_truss(1)_from_SVR4,_and_produced_an_strace_that
worked_on_both_platforms.__(In_1994_Rick_ported_strace_to_SVR4_and
Solaris_and_wrote_the_automatic_configuration_support.__(In_1995
he_ported_strace_to_Irix_and_became_tired_of_writing_about

```

himself_in_the_third_person.

Beginning_with_1996,_strace_was_maintained_by_Wichert_Akkerman.
During_his_tenure,_strace_development_migrated_to_CVS;_ports_to
FreeBSD_and_many_architectures_on_Linux_(including_ARM,_IA-64,
MIPS,_PA-RISC,_PowerPC,_s390,_SPARC)_were_introduced._In_2002,
the_burden_of_strace_maintainership_was_transferred_to_Roland
McGrath._Since_then,_strace_gained_support_for_several_new_Linux
architectures_(AMD64,_s390x,_SuperH),_bi-architecture_support_for
some_of_them,_and_received_numerous_additions_and_improvements_in
syscalls_decoders_on_Linux;_strace_development_migrated_to_Git
during_that_period._Since_2009,_strace_is_actively_maintained_by
Dmitry_Levin._strace_gained_support_for_AArch64,_ARC,_AVR32,
Blackfin,_Meta,_Nios_II,_OpenRISC_1000,_RISC-V,_Tile/TileGx,
Xtensa_architectures_since_that_time._In_2012,_unmaintained_and
apparently_broken_support_for_non-Linux_operating_systems_was
removed._Also,_in_2012_strace_gained_support_for_path_tracing
and_file_descriptor_path_decoding._In_2014,_support_for_stack
trace_printing_was_added._In_2016,_syscall_fault_injection_was
implemented.

For_the_additional_information,_please_refer_to_the_NEWS_file_and
strace_repository_commit_log.

REPORTING_BUGS

Problems_with_strace_should_be_reported_to_the_strace_mailing
list_mailto:strace-devel@lists.strace.io.

SEE_ALSO

strace-log-merge(1),_ltrace(1),_perf-trace(1),_trace-cmd(1),
time(1),_ptrace(2),_seccomp(2),_syscall(2),_proc(5),_signal(7)

strace_Home_Page_https://strace.io/

AUTHORS

The_complete_list_of_strace_contributors_can_be_found_in_the
CREDITS_file.

COLOPHON

This_page_is_part_of_the_strace_(system_call_tracer)_project.
Information_about_the_project_can_be_found_at
<http://strace.io/>._If_you_have_a_bug_report_for_this_manual
page,_send_it_to_strace-devel@lists.sourceforge.net._This_page
was_obtained_from_the_project's_upstream_Git_repository
<https://github.com/strace/strace.git>_on_2024-06-14._(At_that
time,_the_date_of_the_most_recent_commit_that_was_found_in_the
repository_was_2024-06-04.)_If_you_discover_any_rendering
problems_in_this_HTML_version_of_the_page,_or_you_believe_there
is_a_better_or_more_up-to-date_source_for_the_page,_or_you_have
corrections_or_improvements_to_the_information_in_this_COLOPHON
(which_is_not_part_of_the_original_manual_page),_send_a_mail_to
man-pages@man7.org

strace_6.9.0.16.2a4c4_____2024-06-04_____STRACE(1)

3.9 strings: Print Sequences Of Printable Characters

NAME

strings - print the sequences of printable characters **in** files

SYNOPSIS

```
strings [-afovV] [-min-len]
        [-n min-len] [--bytes=min-len]
        [-t radix] [--radix=radix]
        [-e encoding] [--encoding=encoding]
        [-U method] [--unicode=method]
        [-] [--all] [--print-file-name]
        [-T bfdname] [--target=bfdname]
        [-w] [--include-all-whitespace]
        [-s] [--output-separator sep_string]
        [--help] [--version] file...
```

DESCRIPTION

For each file given, GNU strings prints the printable character sequences that are at least 4 characters long (or the number given with the options below) and are followed by an unprintable character.

Depending upon how the strings program was configured it will default to either displaying all the printable sequences that it can find **in** each file, or only those sequences that are **in** loadable, initialized data sections. If the file **type** is unrecognizable, or **if** strings is reading from stdin **then** it will always display all of the printable sequences that it can find.

For backwards compatibility any file that occurs after a **command-**line option of just - will also be scanned **in** full, regardless of the presence of any -d option.

strings is mainly useful **for** determining the contents of non-text files.

OPTIONS

-a
--all
- Scan the whole file, regardless of what sections it contains or whether those sections are loaded or initialized. Normally this is the default behaviour, but strings can be configured so that the -d is the default instead.

The - option is position dependent and forces strings to perform full scans of any file that is mentioned after the - on the **command** line, even **if** the -d option has been specified.

-d
--data
Only print strings from initialized, loaded data sections **in**

the file. This may reduce the amount of garbage **in** the output, but it also exposes the strings program to any security flaws that may be present **in** the BFD library used to scan and load sections. Strings can be configured so that this option is the default behaviour. In such cases the **-a** option can be used to avoid using the BFD library and instead just print all of the strings found **in** the file.

-f
--print-file-name
Print the name of the file before each string.

--help
Print a summary of the program usage on the standard output and exit.

-min-len
-n min-len
--bytes=min-len
Print sequences of displayable characters that are at least min-len characters long. If not specified a default minimum length of 4 is used. The distinction between displayable and non-displayable characters depends upon the setting of the **-e** and **-U** options. Sequences are always terminated at control characters such as new-line and carriage-**return**, but not the tab character.

-o Like **-t o**. Some other versions of strings have **-o** act like **-t d** instead. Since we can not be compatible with both ways, we simply chose one.

-t radix
--radix=radix
Print the offset within the file before each string. The single character argument specifies the radix of the offset---**o** **for** octal, **x** **for** hexadecimal, or **d** **for** decimal.

-e encoding
--encoding=encoding
Select the character encoding of the strings that are to be found. Possible values **for** encoding are: **s** = single-7-bit-byte characters (default), **S** = single-8-bit-byte characters, **b** = 16-bit bigendian, **l** = 16-bit littleendian, **B** = 32-bit bigendian, **L** = 32-bit littleendian. Useful **for** finding wide character strings. (**l** and **b** apply to, **for** example, Unicode UTF-16/UCS-2 encodings).

-U [d|i|l|e|x|h]
--unicode=[default|invalid|locale|escape|hex|highlight]
Controls the display of UTF-8 encoded multibyte characters **in**

strings. The default (`--unicode=default`) is to give them no special treatment, and instead rely upon the setting of the `--encoding` option. The other values **for** this option automatically **enable** `--encoding=S`.

The `--unicode=invalid` option treats them as non-graphic characters and hence not part of a valid string. All the remaining options treat them as valid string characters.

The `--unicode=locale` option displays them **in** the current locale, which may or may not support UTF-8 encoding. The `--unicode=hex` option displays them as hex byte sequences enclosed between `<>` characters. The `--unicode=escape` option displays them as escape sequences (`\uxxxx`) and the `--unicode=highlight` option displays them as escape sequences highlighted **in** red (**if** supported by the output device). The colouring is intended to draw attention to the presence of unicode sequences where they might not be expected.

`-T bfdname`

`--target=bfdname`

Specify an object code format other than your system's default format.

`-v`

`-V`

`--version`

Print the program version number on the standard output and exit.

`-w`

`--include-all-whitespace`

By default tab and space characters are included **in** the strings that are displayed, but other whitespace characters, such as newlines and carriage returns, are not. The `-w` option changes this so that all whitespace characters are considered to be part of a string.

`-s`

`--output-separator`

By default, output strings are delimited by a new-line. This option allows you to supply any string to be used as the output record separator. Useful with `--include-all-whitespace` where strings may contain new-lines internally.

`@file`

Read **command**-line options from file. The options **read** are inserted **in** place of the original `@file` option. If file does not exist, or cannot be **read**, **then** the option will be treated

literally, and not removed.

Options **in** file are separated by whitespace. A whitespace character may be included **in** an option by surrounding the entire option **in** either single or double quotes. Any character (including a backslash) may be included by prefixing the character to be included with a backslash. The file may itself contain additional @file options; any such options will be processed recursively.

SEE ALSO

ar(1), nm(1), objdump(1), ranlib(1), readelf(1) and the Info entries **for** binutils.

COPYRIGHT

Copyright (c) 1991-2024 Free Software Foundation, Inc.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Invariant Sections, with no Front-Cover Texts, and with no Back-Cover Texts. A copy of the license is included **in** the section entitled "GNU_Free_Documentation License".

COLOPHON

This page is part of the binutils (a collection of tools **for** working with executable binaries) project. Information about the project can be found at <http://www.gnu.org/software/binutils/>. If you have a bug report **for** this manual page, see http://sourceware.org/bugzilla/enter_bug.cgi?product=binutils. This page was obtained from the tarball binutils-2.42.tar.gz fetched from <https://ftp.gnu.org/gnu/binutils/> on 2024-06-14. If you discover any rendering problems **in** this HTML version of the page, or you believe there is a better or more up-to-date **source for** the page, or you have corrections or improvements to the information **in** this COLOPHON (which is not part of the original manual page), send a mail to man-pages@man7.org

binutils-2.42

2024-06-14

STRINGS(1)