

**NAME**

**procstat** - get detailed process information

**SYNOPSIS**

**procstat** [--libxo] [-h] [-M *core*] [-N *system*] [-w *interval*] *command* [*pid* ... | *core* ...]

**procstat** [--libxo] -a [-h] [-M *core*] [-N *system*] [-w *interval*] *command*

**procstat** [--libxo] [-h] [-M *core*] [-N *system*] [-w *interval*] [-b | -c | -e | -f [-C] | -i [-n] | -j [-n] | -k [-k] | -l | -r [-H] | -s | -S | -t | -v | -x] [*pid* ... | *core* ...]

**procstat** [--libxo] -a [-h] [-M *core*] [-N *system*] [-w *interval*] [-b | -c | -e | -f [-C] | -i [-n] | -j [-n] | -k [-k] | -l | -r [-H] | -s | -S | -t | -v | -x]

**procstat** [--libxo] -L [-h] [-M *core*] [-N *system*] [-w *interval*] *core* ...

**pargs** [--libxo] *pid* ...

**penv** [--libxo] *pid* ...

**pwdx** [--libxo] *pid* ...

**DESCRIPTION**

**procstat** utility displays detailed information about the processes identified by the *pid* arguments, or if the **-a** flag is used, all processes. It can also display information extracted from a process core file, if the core file is specified as the argument.

The **pargs**, **penv** and **pwdx** utilities display the arguments, environment, and current working directory, respectively of the process specified by *pid* argument. They mimic the behavior of Solaris utilities of the same names.

If the **--libxo** flag is specified the output is generated via libxo(3) in a selection of different human and machine readable formats. See `xo_parse_args(3)` for details on command line arguments.

The following commands are available for **procstat**:

*advlock*

Print information about advisory locks on files. All three types of locks are listed, BSD-style lockf(2), POSIX-style fcntl(2) *F\_SETLK*, and remote lockd(8) locks used by NFSv3.

Note that neither the **-a** option nor *pid* list can be used to limit the display of the locks, mostly because some types of locks do not have local (or any) owning processes.

*basic* Print basic process statistics (this is the default).

*binary* | **-b**

Display binary information for the process.

Substring commands are accepted.

*argument(s)* | **-c**

Display command line arguments for the process.

Substring commands are accepted.

*environment* | **-e**

Display environment variables for the process.

Substring commands are accepted.

*file(s)* | *fd(s)* | **-f**

Display file descriptor information for the process.

If the **-C** subcommand flag is used then additional capability information is printed.

*signal(s)* | **-i**

Display signal pending and disposition information for the process.

If the **-n** subcommand option is used, the signal numbers are shown instead of signal names.

Substring commands are accepted.

*tsignal(s)* | **-j**

Display signal pending and blocked information for the process's threads.

If the **-n** subcommand option is used, the signal numbers are shown instead of signal names.

Substring commands are accepted.

*kstack* | **-k**

Display the stacks of kernel threads in the process, excluding stacks of threads currently running on a CPU and threads with stacks swapped to disk.

If the **-v** subcommand option is used (or the command flag is repeated), function offsets as well as function names are printed.

*rlimit* | **-l**

Display resource limits for the process.

*ptlwpinfo* | **-L**

Display LWP info for the process pertaining to its signal driven exit.

*rusage* | **-r**

Display resource usage information for the process.

If the **-v** (or **-H**) subcommand flag is used then per-thread statistics are printed, rather than per-process statistics. The second field in the table will list the thread ID to which the row of information corresponds.

*credential(s)* | **-s**

Display security credential information for the process.

Substring commands are accepted.

*cpuset* | *cs* | **-S**

Display the cpuset information for the thread.

*thread(s)* | **-t**

Display thread information for the process.

*vm* | **-v**

Display virtual memory mappings for the process.

*auxv* | **-x**

Display ELF auxiliary vector for the process.

*pargs* Display arguments for the process.

*penv* Display environment variables for the process.

*pwdx* Display current working directory for the process.

All options generate output in the format of a table, the first field of which is the process ID to which the row of information corresponds. The **-h** flag may be used to suppress table headers.

The **-w** flag may be used to specify a wait interval at which to repeat the printing of the requested process information. If the **-w** flag is not specified, the output will not repeat.

Information for VM, file descriptor, and cpuset options is available only to the owner of a process or the

superuser. A cpuset value displayed as -1 means that the information is either invalid or not available.

### Binary Information

Display the process ID, command, and path to the process binary:

PID process ID  
COMM  
command  
OSREL  
osreldate for process binary  
PATH  
path to process binary (if available)

### Command Line Arguments

Display the process ID, command, and command line arguments:

PID process ID  
COMM  
command  
ARGS  
command line arguments (if available)

### Environment Variables

Display the process ID, command, and environment variables:

PID process ID  
COMM command  
ENVIRONMENT environment variables (if available)

### File Descriptors

Display detailed information about each file descriptor referenced by a process, including the process ID, command, file descriptor number, and per-file descriptor object information, such as object type and file system path. By default, the following information will be printed:

PID process ID  
COMM  
command  
FD file descriptor number or cwd/root/jail  
T file descriptor type  
V vnode type

**FLAGS**

file descriptor flags

**REF** file descriptor reference count

**OFFSET**

file descriptor offset

**PRO** network protocol

**NAME**

file path or socket addresses (if available)

The following file descriptor types may be displayed:

e POSIX semaphore

E eventfd

f fifo

h shared memory

k kqueue

m

message queue

P process descriptor

p pipe

s socket

t pseudo-terminal master

v vnode

The following vnode types may be displayed:

- not a vnode

b block device

c character device

d directory

f fifo

l symbolic link

r regular file

s socket

x revoked device

The following file descriptor flags may be displayed:

r read

w write

a append  
 s async  
 f fsync  
 n non-blocking  
 d direct I/O  
 l lock held

If the **-C** flag is specified, the vnode type, reference count, and offset fields will be omitted, and a new capabilities field will be included listing capabilities, as described in `cap_rights_limit(2)`, present for each capability descriptor.

The following network protocols may be displayed (grouped by address family):

AF\_INET, AF\_INET6

ICM IPPROTO\_ICMP; see `icmp(4)`.  
 IP? unknown protocol.  
 RAW IPPROTO\_RAW; see `ip(4)`.  
 SCT IPPROTO\_SCTP; see `sctp(4)`.  
 TCP IPPROTO\_TCP; see `tcp(4)`.  
 UDP IPPROTO\_UDP; see `udp(4)`.

AF\_LOCAL

UDD IPPROTO\_UDP; see `udp(4)`.  
 UDS IPPROTO\_TCP; see `tcp(4)`.  
 UD? unknown protocol.

AF\_DIVERT

IPD Divert socket; see `divert(4)`.  
 ? unknown address family.

### Signal Disposition Information

Display signal pending and disposition for a process:

PID process ID  
 COMM  
 command

SIG signal name

FLAGS

process signal disposition details, three symbols

P if signal is pending in the global process queue; - otherwise.

I if signal delivery disposition is SIG\_IGN; - otherwise.

C if the signal will be caught; - otherwise.

If **-n** switch is given, the signal numbers are shown instead of signal names.

### Thread Signal Information

Display signal pending and blocked for a process's threads:

PID process ID

TID thread ID

COMM

command

SIG signal name

FLAGS

thread signal delivery status, two symbols

P if signal is pending for the thread, - otherwise

B if signal is blocked in the thread signal mask, - if not blocked

The **-n** switch has the same effect as for the **-i** switch: the signal numbers are shown instead of signal names.

### Kernel Thread Stacks

Display kernel thread stacks for a process, allowing further interpretation of thread wait channels. If the **-k** flag is repeated, function offsets, not just function names, are printed.

This feature requires **options STACK** or **options DDB** to be compiled into the kernel.

PID process ID

TID thread ID

COMM

command

TDNAME

thread name

KSTACK

kernel thread call stack

**Resource Limits**

Display resource limits for a process:

PID process ID  
COMM  
command  
RLIMIT  
resource limit name  
SOFT soft limit  
HARD  
hard limit

**Resource Usage**

Display resource usage for a process. If the **-H** flag is specified, resource usage for individual threads is displayed instead.

PID process ID  
TID thread ID (if **-H** is specified)  
COMM command  
RESOURCE resource name  
VALUE current usage

**Security Credentials**

Display process credential information:

PID process ID  
COMM  
command  
EUID effective user ID  
RUID real user ID  
SVUID  
saved user ID  
EGID effective group ID  
RGID real group ID  
SVGID  
saved group ID  
UMASK  
file creation mode mask  
FLAGS  
credential flags

**GROUPS**

group set

The following credential flags may be displayed:

C capability mode

**Thread Information**

Display per-thread information, including process ID, per-thread ID, name, CPU, and execution state:

PID process ID

TID thread ID

COMM

command

TDNAME

thread name

CPU current or most recent CPU run on

PRI thread priority

STATE

thread state

WCHAN

thread wait channel

**Virtual Memory Mappings**

Display process virtual memory mappings, including addresses, mapping meta-data, and mapped object information:

PID process ID

START

starting address of mapping

END ending address of mapping

PRT protection flags

RES resident pages

PRES private resident pages

REF reference count

SHD shadow page count

FLAG

mapping flags

TP VM object type

The following protection flags may be displayed:

r read  
w write  
x execute

The following VM object types may be displayed:

-- none  
dd dead  
df default  
dv device  
md device with managed pages (GEM/TTM)  
ph physical  
sg scatter/gather  
sw swap  
vn vnode  
gd guard (pseudo-type)

The following mapping flags may be displayed:

C copy-on-write  
N needs copy  
S one or more superpage mappings are used  
D grows down (top-down stack)  
U grows up (bottom-up stack)  
W  
    pages in this range are locked by mlock(2) or mlockall(2)

### **ELF Auxiliary Vector**

Display ELF auxiliary vector values:

PID process ID  
COMM  
    command  
AUXV  
    auxiliary vector name  
VALUE  
    auxiliary vector value

**Advisory Lock Information**

RW Read/Write type, *RO* for read, *RW* for write lock

TYPE Type of the lock, one of *FLOCK* for flock(2), *FCNTL* for fcntl(2), *LOCKD* for remote

PID Process id of the owner, for *FCNTL* and remote types

SYSID

Remote system id if applicable

FSID File system id where the locked file reside

RDEV

rdev for the file system

INO Unique file identifier (inode number) of the locked file on the file system

START

Start offset of the locked range

LEN Length of the locked range. Zero means till EOF

PATH

If available, the path of the locked file

**EXIT STATUS**

The **procstat** utility exits 0 on success, and >0 if an error occurs.

**EXAMPLES**

Show binary information about the current shell:

```
$ procstat binary $$
  PID COMM      OSREL PATH
  46620 bash    1201000 /usr/local/bin/bash
```

Same as above but showing information about open file descriptors:

```
$ procstat files $$
  PID COMM      FD T V FLAGS  REF  OFFSET PRO NAME
  46620 bash    text v r r----- -  - - /usr/local/bin/bash
  46620 bash    ctty v c rw----- -  - - /dev/pts/12
  46620 bash    cwd v d r----- -  - - /tmp
  46620 bash    root v d r----- -  - - /
  46620 bash    0 v c rw----- 7 372071 - /dev/pts/12
  46620 bash    1 v c rw----- 7 372071 - /dev/pts/12
  46620 bash    2 v c rw----- 7 372071 - /dev/pts/12
  46620 bash    255 v c rw----- 7 372071 - /dev/pts/12
```

Show the arguments used to launch init(8):

```
$ procstat arguments 1
PID COMM      ARGS
  1 init      /sbin/init --
```

Extract binary information from a core dump:

```
$ procstat binary core.36642
PID COMM      OSREL PATH
36642 top      1201000 /usr/bin/top
```

Trying to extract information from a core file generated in a different major FreeBSD version might show an error like this:

```
$ procstat mplayer.core
procstat: kinfo_proc structure size mismatch
procstat: procstat_getprocs()
```

## SEE ALSO

fstat(1), ps(1), sockstat(1), cap\_enter(2), cap\_rights\_limit(2), mlock(2), mlockall(2), libprocstat(3), libxo(3), signal(3), xo\_parse\_args(3), ddb(4), divert(4), icmp(4), ip(4), sctp(4), tcp(4), udp(4), stack(9)

## AUTHORS

Robert N M Watson <[rwatson@FreeBSD.org](mailto:rwatson@FreeBSD.org)>.  
 libxo(3) support was added by Allan Jude <[allanjude@FreeBSD.org](mailto:allanjude@FreeBSD.org)>.  
 Juraj Lutter <[juraj@lutter.sk](mailto:juraj@lutter.sk)> added the pargs, penv and pwdx functionality.

## BUGS

The display of open file or memory mapping pathnames is implemented using the kernel's name cache. If a file system does not use the name cache, or the path to a file is not in the cache, a path will not be displayed.

**procstat** currently supports extracting data only from a live kernel, and not from kernel crash dumps.