NAME

procfs - process file system

SYNOPSIS

```
proc                    /proc       procfs      rw 0 0
```

DESCRIPTION

This functionality is deprecated. Users are advised to use libprocstat(3) and kvm(3) instead.

The process file system, or **procfs**, implements a view of the system process table inside the file system. It is normally mounted on */proc*.

The **procfs** provides a two-level view of process space, unlike the previous FreeBSD 1.1 **procfs** implementation. At the highest level, processes themselves are named, according to their process ids in decimal, with no leading zeros. There is also a special node called *curproc* which always refers to the process making the lookup request.

Each node is a directory which contains the following entries:

dbregs
: The debug registers as defined by struct dbregs in <*machine/reg.h*>. *dbregs* is currently only implemented on the i386 architecture.

etype
: The type of the executable referenced by the *file* entry.

file
: A symbolic link to the file from which the process text was read. This can be used to gain access to the process' symbol table, or to start another copy of the process. If the file cannot be found, the link target is 'unknown'.

fpregs
: The floating point registers as defined by struct fpregs in <*machine/reg.h*>. *fpregs* is only implemented on machines which have distinct general purpose and floating point register sets.

map
: A collection of lines describing the memory regions of the process, where each line contains the following fields:

start-address	The starting address for the region (inclusive).
end-address	The ending address for the region (exclusive).
resident	The number of resident pages.
private-resident	The number of resident pages that were private to the process.
obj	The virtual address of the *struct vm_object* kernel data structure describing the

memory region.

access — A three character string comprising the characters ‘r’, ‘w’ and ‘x’, denoting read, write, and execute permissions respectively. The lack of a permission is represented by ‘-’.

ref_count — The number of references to the region.

shadow_count — The number of VM objects that this region is a shadow for.

flags — The flags for the object, see the flags named **OBJ_*** in <*vm/vm_object.h*>.

copy-on-write — Whether the region is copy-on-write. One of:

- COW A copy-on-write region.
- NCOW A non-copy-on-write region.

needs-copy — Whether the region needs a copy. One of:

- NC The region needs a copy.
- NNC The region does not need a copy.

type — The type of the region. One of:

- dead A region associated with a dead VM object.
- device A region backed by device memory.
- none A region not backed by anything.
- phys A region backed by physical memory.
- swap A region backed by swap.
- unknown A region of unknown type.
- vnode A region backed by a file.

fullpath — The path to the file backing the memory region, or ‘-’ if there is no such file.

cred — One of:

- CH The region is being charged to the user specified in the ‘charged-uid’ field.
- NCH The region is not being charged to any user.

charged-uid — The UID of the user being charged, or -1 if no user is being charged.

mem The complete virtual memory image of the process. Only those address which exist in the process can be accessed. Reads and writes to this file modify the process. Writes to the text segment remain private to the process.

note Used for sending signals to the process. Not implemented.

notepg
Used for sending signal to the process group. Not implemented.

osrel Allows read and write of the kernel osrel value assigned to the process. It affects the compatibility shims that are turned on and off depending on the value. Initial process value is read from the ABI note tag in the executed ELF image, and is zero if the tag not supported by

binary format or was not found.

regs Allows read and write access to the process' register set. This file contains a binary data structure struct regs defined in <*machine/reg.h*>. *regs* can only be written when the process is stopped.

rlimit This is a read-only file containing the process current and maximum limits. Each line is of the format *rlimit current max*, with -1 indicating infinity.

status The process status. This file is read-only and returns a single line containing multiple space-separated fields as follows:

- command name
- process id
- parent process id
- process group id
- session id
- device name of the controlling terminal, or a minus sign ("-") if there is no controlling terminal.
- a list of process flags: ctty if there is a controlling terminal, sldr if the process is a session leader, noflags if neither of the other two flags are set.
- the process start time in seconds and microseconds, comma separated.
- the user time in seconds and microseconds, comma separated.
- the system time in seconds and microseconds, comma separated.
- the wait channel message
- the process credentials consisting of the effective user id and the list of groups (whose first member is the effective group id) all comma separated.
- the hostname of the jail in which the process runs, or '-' to indicate that the process is not running within a jail.

Each node is owned by the process's user, and belongs to that user's primary group.

FILES

/proc	normal mount point for the **procfs**.
/proc/pid	directory containing process information for process *pid*.
/proc/curproc	directory containing process information for the current process
/proc/curproc/cmdline	the process executable name
/proc/curproc/etype	executable type
/proc/curproc/file	executable image
/proc/curproc/fpregs	the process floating point register set
/proc/curproc/map	virtual memory map of the process

/proc/curproc/mem	the complete virtual address space of the process
/proc/curproc/note	used for signaling the process
/proc/curproc/notepg	used for signaling the process group
/proc/curproc/osrel	the process osrel value
/proc/curproc/regs	the process register set
/proc/curproc/rlimit	the process current and maximum rlimit
/proc/curproc/status	the process' current status

EXAMPLES

To mount a **procfs** file system on */proc*:

```
mount -t procfs proc /proc
```

SEE ALSO

procstat(1), mount(2), sigaction(2), unmount(2), kvm(3), libprocstat(3), pseudofs(9)

AUTHORS

This manual page written by Garrett Wollman, based on the description provided by Jan-Simon Pendry, and revamped later by Mike Pritchard.