

NAME

kvm_getprocs, **kvm_getargv**, **kvm_getenvv** - access user process state

LIBRARY

Kernel Data Access Library (libkvm, -lkvm)

SYNOPSIS

```
#include <kvm.h>
```

```
#include <sys/param.h>
```

```
#include <sys/sysctl.h>
```

```
#include <sys/user.h>
```

```
struct kinfo_proc *
```

```
kvm_getprocs(kvm_t *kd, int op, int arg, int *cnt);
```

```
char **
```

```
kvm_getargv(kvm_t *kd, const struct kinfo_proc *p, int nchr);
```

```
char **
```

```
kvm_getenvv(kvm_t *kd, const struct kinfo_proc *p, int nchr);
```

DESCRIPTION

The **kvm_getprocs()** function returns a (sub-)set of active processes in the kernel indicated by *kd*. The *op* and *arg* arguments constitute a predicate which limits the set of processes returned. The value of *op* describes the filtering predicate as follows:

KERN_PROC_ALL all processes and kernel visible threads

KERN_PROC_PROC all processes, without threads

KERN_PROC_PID processes with process ID *arg*

KERN_PROC_PGRP processes with process group *arg*

KERN_PROC_SESSION
 processes with session *arg*

KERN_PROC_TTY processes with TTY *arg*

KERN_PROC_UID processes with effective user ID *arg*

KERN_PROC_RUID processes with real user ID *arg*

KERN_PROC_INC_THREAD

 modifier to return all kernel visible threads when filtering by process ID, process group, TTY, user ID, and real user ID

The number of processes found is returned in the reference parameter *cnt*. The processes are returned as

a contiguous array of `kinfo_proc` structures. This memory is locally allocated, and subsequent calls to `kvm_getprocs()` and `kvm_close()` will overwrite this storage.

The `kvm_getargv()` function returns a null-terminated argument vector that corresponds to the command line arguments passed to process indicated by `p`. Most likely, these arguments correspond to the values passed to `exec(3)` on process creation. This information is, however, deliberately under control of the process itself. Note that the original command name can be found, unaltered, in the `p_comm` field of the process structure returned by `kvm_getprocs()`.

The `nchr` argument indicates the maximum number of characters, including null bytes, to use in building the strings. If this amount is exceeded, the string causing the overflow is truncated and the partial result is returned. This is handy for programs like `ps(1)` and `w(1)` that print only a one line summary of a command and should not copy out large amounts of text only to ignore it. If `nchr` is zero, no limit is imposed and all argument strings are returned in their entirety.

The memory allocated to the `argv` pointers and string storage is owned by the `kvm` library. Subsequent `kvm_getprocs()` and `kvm_close(3)` calls will clobber this storage.

The `kvm_getenvv()` function is similar to `kvm_getargv()` but returns the vector of environment strings. This data is also alterable by the process.

RETURN VALUES

The `kvm_getprocs()`, `kvm_getargv()`, and `kvm_getenvv()` functions return `NULL` on failure.

SEE ALSO

`kvm(3)`, `kvm_close(3)`, `kvm_geterr(3)`, `kvm_nlist(3)`, `kvm_open(3)`, `kvm_openfiles(3)`, `kvm_read(3)`, `kvm_write(3)`

BUGS

These routines do not belong in the `kvm` interface.