NAME

kvm - kernel memory interface

LIBRARY

Kernel Data Access Library (libkvm, -lkvm)

DESCRIPTION

The **kvm** library provides a uniform interface for accessing kernel virtual memory images, including live systems and crash dumps. Access to live systems is via sysctl(3) for some functions, and mem(4) and kmem(4) for other functions, while crash dumps can be examined via the core file generated by savecore(8). The interface behaves similarly in both cases. Memory can be read and written, kernel symbol addresses can be looked up efficiently, and information about user processes can be gathered.

The **kvm_open**() function is first called to obtain a descriptor for all subsequent calls.

COMPATIBILITY

The kvm interface was first introduced in SunOS. A considerable number of programs have been developed that use this interface, making backward compatibility highly desirable. In most respects, the Sun kvm interface is consistent and clean. Accordingly, the generic portion of the interface (i.e., **kvm_open**(), **kvm_close**(), **kvm_read**(), **kvm_write**(), and **kvm_nlist**()) has been incorporated into the BSD interface. Indeed, many kvm applications (i.e., debuggers and statistical monitors) use only this subset of the interface.

The process interface was not kept. This is not a portability issue since any code that manipulates processes is inherently machine dependent.

Finally, the Sun kvm error reporting semantics are poorly defined. The library can be configured either to print errors to stderr automatically, or to print no error messages at all. In the latter case, the nature of the error cannot be determined. To overcome this, the BSD interface includes a routine, kvm_geterr(3), to return (not print out) the error message corresponding to the most recent error condition on the given descriptor.

CROSS DEBUGGING

The **kvm** library supports inspection of crash dumps from non-native kernels. Only a limited subset of the kvm interface is supported for these dumps. To inspect a crash dump of a non-native kernel, the caller must provide a *resolver* function when opening a descriptor via **kvm_open2**(). In addition, the kvm interface defines an integer type (*kvaddr_t*) that is large enough to hold all valid addresses of all supported architectures. The interface also defines a new namelist structure type (*struct kvm_nlist*) for use with **kvm_nlist2**(). To avoid address truncation issues, the caller should use **kvm_nlist2**() and **kvm_read2**() in place of **kvm_nlist**() and **kvm_read**(), respectively. Finally, only a limited subset of operations are supported for non-native crash dumps: **kvm_close**(), **kvm_geterr**(), **kvm_kerndisp**(), **kvm_open2**(), **kvm_native**(), **kvm_nlist2**(), and **kvm_read2**().

SEE ALSO

kvm_close(3), kvm_getargv(3), kvm_getenvv(3), kvm_geterr(3), kvm_getloadavg(3), kvm_getprocs(3), kvm_getswapinfo(3), kvm_kerndisp(3), kvm_native(3), kvm_nlist(3), kvm_nlist2(3), kvm_open(3), kvm_open2(3), kvm_openfiles(3), kvm_read(3), kvm_read2(3), kvm_write(3), sysctl(3), kmem(4), mem(4)

HISTORY

The **kvm_native**(), **kvm_nlist2**(), **kvm_open2**(), and **kvm_read2**() functions first appeared in FreeBSD 11.0.