NAME

query_module − query the kernel for various bits pertaining to modules

SYNOPSIS

#include <linux/module.h>

[[deprecated]] int query_module(const char ***_name_, int** _which_**,**
void _buf_**[.**_bufsize_**], size_t** _bufsize_**,**
size_t ***_ret_);**

DESCRIPTION

Note: This system call is present only before Linux 2.6.

query_module() requests information from the kernel about loadable modules. The returned information is placed in the buffer pointed to by _buf_. The caller must specify the size of _buf_ in _bufsize_. The precise nature and format of the returned information depend on the operation specified by _which_. Some operations require _name_ to identify a currently loaded module, some allow _name_ to be NULL, indicating the kernel proper.

The following values can be specified for _which_:

0 Returns success, if the kernel supports **query_module**(). Used to probe for availability of the system call.

QM_MODULES
Returns the names of all loaded modules. The returned buffer consists of a sequence of null-terminated strings; _ret_ is set to the number of modules.

QM_DEPS
Returns the names of all modules used by the indicated module. The returned buffer consists of a sequence of null-terminated strings; _ret_ is set to the number of modules.

QM_REFS
Returns the names of all modules using the indicated module. This is the inverse of **QM_DEPS**. The returned buffer consists of a sequence of null-terminated strings; _ret_ is set to the number of modules.

QM_SYMBOLS
Returns the symbols and values exported by the kernel or the indicated module. The returned buffer is an array of structures of the following form

```
struct module_symbol {
    unsigned long value;
    unsigned long name;
};
```

followed by null-terminated strings. The value of _name_ is the character offset of the string relative to the start of _buf_; _ret_ is set to the number of symbols.

QM_INFO
Returns miscellaneous information about the indicated module. The output buffer format is:

```
struct module_info {
    unsigned long address;
    unsigned long size;
    unsigned long flags;
};
```

where _address_ is the kernel address at which the module resides, _size_ is the size of the module in bytes, and _flags_ is a mask of **MOD_RUNNING**, **MOD_AUTOCLEAN**, and so on, that indicates the current status of the module (see the Linux kernel source file _include/linux/module.h_). _ret_ is set to the size of the _module_info_ structure.

RETURN VALUE

On success, zero is returned. On error, −1 is returned and *errno* is set to indicate the error.

ERRORS

EFAULT
At least one of *name*, *buf*, or *ret* was outside the program's accessible address space.

EINVAL
Invalid *which*; or *name* is NULL (indicating "the kernel"), but this is not permitted with the specified value of *which*.

ENOENT
No module by that *name* exists.

ENOSPC
The buffer size provided was too small. *ret* is set to the minimum size needed.

ENOSYS
query_module() is not supported in this version of the kernel (e.g., Linux 2.6 or later).

VERSIONS

This system call is present only up until Linux 2.4; it was removed in Linux 2.6.

STANDARDS

query_module() is Linux-specific.

NOTES

Some of the information that was formerly available via **query_module**() can be obtained from */proc/modules*, */proc/kallsyms*, and the files under the directory */sys/module*.

The **query_module**() system call is not supported by glibc. No declaration is provided in glibc headers, but, through a quirk of history, glibc does export an ABI for this system call. Therefore, in order to employ this system call, it is sufficient to manually declare the interface in your code; alternatively, you can invoke the system call using **syscall**(2).

SEE ALSO

create_module(2), **delete_module**(2), **get_kernel_syms**(2), **init_module**(2), **lsmod**(8), **modinfo**(8)