NAME

sysctl − read/write system parameters

SYNOPSIS

#include <unistd.h>
#include <linux/sysctl.h>

[[deprecated]] int _sysctl(struct __sysctl_args **args*);

DESCRIPTION

This system call no longer exists on current kernels! See NOTES.

The **_sysctl**() call reads and/or writes kernel parameters. For example, the hostname, or the maximum number of open files. The argument has the form

```
struct __sysctl_args {
    int    *name;    /* integer vector describing variable */
    int     nlen;    /* length of this vector */
    void   *oldval;  /* 0 or address where to store old value */
    size_t *oldlenp; /* available room for old value,
                        overwritten by actual size of old value */
    void   *newval;  /* 0 or address of new value */
    size_t  newlen;  /* size of new value */
};
```

This call does a search in a tree structure, possibly resembling a directory tree under */proc/sys*, and if the requested item is found calls some appropriate routine to read or modify the value.

RETURN VALUE

Upon successful completion, **_sysctl**() returns 0. Otherwise, a value of −1 is returned and *errno* is set to indicate the error.

ERRORS

EACCES, **EPERM**
: No search permission for one of the encountered "directories", or no read permission where *oldval* was nonzero, or no write permission where *newval* was nonzero.

EFAULT
: The invocation asked for the previous value by setting *oldval* non-NULL, but allowed zero room in *oldlenp*.

ENOTDIR
: *name* was not found.

VERSIONS

This system call first appeared in Linux 1.3.57. It was removed in Linux 5.5; glibc support was removed in glibc 2.32.

STANDARDS

This call is Linux-specific, and should not be used in programs intended to be portable. It originated in 4.4BSD. Only Linux has the*/pr oc/sys* mirror, and the object naming schemes differ between Linux and 4.4BSD, but the declaration of the **sysctl**() function is the same in both.

NOTES

Use of this system call was long discouraged: since Linux 2.6.24, uses of this system call result in warnings in the kernel log, and in Linux 5.5, the system call was finally removed. Use the*/pr oc/sys* interface instead.

Note that on older kernels where this system call still exists, it is available only if the kernel was configured with the **CONFIG_SYSCTL_SYSCALL** option. Furthermore, glibc does not provide a wrapper for this system call, necessitating the use of **syscall**(2).

BUGS

The object names vary between kernel versions, making this system call worthless for applications.

Not all available objects are properly documented.

It is not yet possible to change operating system by writing to */proc/sys/kernel/ostype*.

EXAMPLES

```
#define _GNU_SOURCE
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <sys/syscall.h>
#include <unistd.h>

#include <linux/sysctl.h>

#define ARRAY_SIZE(arr)  (sizeof(arr) / sizeof((arr)[0]))

int _sysctl(struct __sysctl_args *args);

#define OSNAMESZ 100

int
main(void)
{
    int                  name[] = { CTL_KERN, KERN_OSTYPE };
    char                 osname[OSNAMESZ];
    size_t               osnamelth;
    struct __sysctl_args  args;

    memset(&args, 0, sizeof(args));
    args.name = name;
    args.nlen = ARRAY_SIZE(name);
    args.oldval = osname;
    args.oldlenp = &osnamelth;

    osnamelth = sizeof(osname);

    if (syscall(SYS__sysctl, &args) == -1) {
        perror("_sysctl");
        exit(EXIT_FAILURE);
    }
    printf("This machine is running %*s\n", (int) osnamelth, osname);
    exit(EXIT_SUCCESS);
}
```

SEE ALSO

proc(5)