

**NAME**

syslog, klogctl - read and/or clear kernel message ring buffer; set console\_loglevel

**SYNOPSIS**

```
int syslog(int type, char *bufp, int len);
/* No wrapper provided in glibc */

/* The glibc interface */
#include <sys/klog.h>

int klogctl(int type, char *bufp, int len);
```

**DESCRIPTION**

*Note:* Probably, you are looking for the C library function `syslog()`, which talks to `syslogd(8)`; see [syslog\(3\)](#) for details.

This page describes the kernel `syslog()` system call, which is used to control the kernel `printk()` buffer; the glibc wrapper function for the system call is called `klogctl()`.

**The kernel log buffer**

The kernel has a cyclic buffer of length `LOG_BUF_LEN` in which messages given as arguments to the kernel function `printk()` are stored (regardless of their log level). In early kernels, `LOG_BUF_LEN` had the value 4096; from kernel 1.3.54, it was 8192; from kernel 2.1.113, it was 16384; since kernel 2.4.23/2.6, the value is a kernel configuration option (`CONFIG_LOG_BUF_SHIFT`, default value dependent on the architecture). Since Linux 2.6.6, the size can be queried with command type 10 (see below).

**Commands**

The *type* argument determines the action taken by this function. The list below specifies the values for *type*. The symbolic names are defined in the kernel source, but are not exported to user space; you will either need to use the numbers, or define the names yourself.

**SYSLOG\_ACTION\_CLOSE** (0)

Close the log. Currently a NOP.

**SYSLOG\_ACTION\_OPEN** (1)

Open the log. Currently a NOP.

**SYSLOG\_ACTION\_READ** (2)

Read from the log. The call waits until the kernel log buffer is nonempty, and then reads at most *len* bytes into the buffer pointed to by *bufp*. The call returns the number of bytes read. Bytes read from the log disappear from the log buffer: the information can be read only once. This is the function executed by the kernel when a user program reads `/proc/kmsg`.

**SYSLOG\_ACTION\_READ\_ALL** (3)

Read all messages remaining in the ring buffer, placing them in the buffer pointed to by *bufp*. The call reads the last *len* bytes from the log buffer (nondestructively), but will not read more than was written into the buffer since the last clear ring buffer command (see command 5 below). The call returns the number of bytes read.

**SYSLOG\_ACTION\_READ\_CLEAR** (4)

Read and clear all messages remaining in the ring buffer. The call does precisely the same as for a *type* of 3, but also executes the clear ring buffer command.

**SYSLOG\_ACTION\_CLEAR** (5)

The call executes just the clear ring buffer command. The *bufp* and *len* arguments are ignored.

This command does not really clear the ring buffer. Rather, it sets a kernel bookkeeping variable that determines the results returned by commands 3 (`SYSLOG_ACTION_READ_ALL`) and 4 (`SYSLOG_ACTION_READ_CLEAR`). This command has no effect on commands 2 (`SYSLOG_ACTION_READ`) and 9

(**SYSLOG\_ACTION\_SIZE\_UNREAD**).

**SYSLOG\_ACTION\_CONSOLE\_OFF** (6)

The command saves the current value of *console\_loglevel* and then sets *console\_loglevel* to *minimum\_console\_loglevel*, so that no messages are printed to the console. Before Linux 2.6.32, the command simply sets *console\_loglevel* to *minimum\_console\_loglevel*. See the discussion of */proc/sys/kernel/printk*, below.

The *bufp* and *len* arguments are ignored.

**SYSLOG\_ACTION\_CONSOLE\_ON** (7)

If a previous **SYSLOG\_ACTION\_CONSOLE\_OFF** command has been performed, this command restores *console\_loglevel* to the value that was saved by that command. Before Linux 2.6.32, this command simply sets *console\_loglevel* to *default\_console\_loglevel*. See the discussion of */proc/sys/kernel/printk*, below.

The *bufp* and *len* arguments are ignored.

**SYSLOG\_ACTION\_CONSOLE\_LEVEL** (8)

The call sets *console\_loglevel* to the value given in *len*, which must be an integer between 1 and 8 (inclusive). The kernel silently enforces a minimum value of *minimum\_console\_loglevel* for *len*. See the *log level* section for details. The *bufp* argument is ignored.

**SYSLOG\_ACTION\_SIZE\_UNREAD** (9) (since Linux 2.4.10)

The call returns the number of bytes currently available to be read from the kernel log buffer via command 2 (**SYSLOG\_ACTION\_READ**). The *bufp* and *len* arguments are ignored.

**SYSLOG\_ACTION\_SIZE\_BUFFER** (10) (since Linux 2.6.6)

This command returns the total size of the kernel log buffer. The *bufp* and *len* arguments are ignored.

All commands except 3 and 10 require privilege. In Linux kernels before 2.6.37, command types 3 and 10 are allowed to unprivileged processes; since Linux 2.6.37, these commands are allowed to unprivileged processes only if */proc/sys/kernel/dmesg\_restrict* has the value 0. Before Linux 2.6.37, privileged means that the caller has the **CAP\_SYS\_ADMIN** capability. Since Linux 2.6.37, privileged means that the caller has either the **CAP\_SYS\_ADMIN** capability (now deprecated for this purpose) or the (new) **CAP\_SYSLOG** capability.

**/proc/sys/kernel/printk**

*/proc/sys/kernel/printk* is a writable file containing four integer values that influence kernel *printk()* behavior when printing or logging error messages. The four values are:

*console\_loglevel*

Only messages with a log level lower than this value will be printed to the console. The default value for this field is **DEFAULT\_CONSOLE\_LOGLEVEL** (7), but it is set to 4 if the kernel command line contains the word *quiet*, 10 if the kernel command line contains the word *debug*, and to 15 in case of a kernel fault (the 10 and 15 are just silly, and equivalent to 8). The value of *console\_loglevel* can be set (to a value in the range 1-8) by a **syslog()** call with a *type* of 8.

*default\_message\_loglevel*

This value will be used as the log level for *printk()* messages that do not have an explicit level. Up to and including Linux 2.6.38, the hard-coded default value for this field was 4 (**KERN\_WARNING**); since Linux 2.6.39, the default value is defined by the kernel configuration option **CONFIG\_DEFAULT\_MESSAGE\_LOGLEVEL**, which defaults to 4.

*minimum\_console\_loglevel*

The value in this field is the minimum value to which *console\_loglevel* can be set.

*default\_console\_loglevel*

This is the default value for *console\_loglevel*.

### The log level

Every *printk()* message has its own log level. If the log level is not explicitly specified as part of the message, it defaults to *default\_message\_loglevel*. The conventional meaning of the log level is as follows:

Kernel constant	Level value	Meaning
<b>KERN_EMERG</b>	0	System is unusable
<b>KERN_ALERT</b>	1	Action must be taken immediately
<b>KERN_CRIT</b>	2	Critical conditions
<b>KERN_ERR</b>	3	Error conditions
<b>KERN_WARNING</b>	4	Warning conditions
<b>KERN_NOTICE</b>	5	Normal but significant condition
<b>KERN_INFO</b>	6	Informational
<b>KERN_DEBUG</b>	7	Debug-level messages

The kernel *printk()* routine will print a message on the console only if it has a log level less than the value of *console\_loglevel*.

### RETURN VALUE

For *type* equal to 2, 3, or 4, a successful call to **syslog()** returns the number of bytes read. For *type* 9, **syslog()** returns the number of bytes currently available to be read on the kernel log buffer. For *type* 10, **syslog()** returns the total size of the kernel log buffer. For other values of *type*, 0 is returned on success.

In case of error, -1 is returned, and *errno* is set to indicate the error.

### ERRORS

#### EINVAL

Bad arguments (e.g., bad *type*; or for *type* 2, 3, or 4, *buf* is NULL, or *len* is less than zero; or for *type* 8, the *level* is outside the range 1 to 8).

#### ENOSYS

This **syslog()** system call is not available, because the kernel was compiled with the **CONFIG\_PRINTK** kernel-configuration option disabled.

#### EPERM

An attempt was made to change *console\_loglevel* or clear the kernel message ring buffer by a process without sufficient privilege (more precisely: without the **CAP\_SYS\_ADMIN** or **CAP\_SYSLOG** capability).

#### ERESTARTSYS

System call was interrupted by a signal; nothing was read. (This can be seen only during a trace.)

### CONFORMING TO

This system call is Linux-specific and should not be used in programs intended to be portable.

### NOTES

From the very start, people noted that it is unfortunate that a system call and a library routine of the same name are entirely different animals.

### SEE ALSO

[syslog\(3\)](#), [capabilities\(7\)](#)

### COLOPHON

This page is part of release 3.74 of the Linux *man-pages* project. A description of the project, information about reporting bugs, and the latest version of this page, can be found at <http://www.kernel.org/doc/man-pages/>.