

**NAME**

closelog, openlog, syslog, vsyslog – send messages to the system logger

**LIBRARY**

Standard C library (*libc*, *-lc*)

**SYNOPSIS**

```
#include <syslog.h>
```

```
void openlog(const char *ident, int option, int facility);
```

```
void syslog(int priority, const char *format, ...);
```

```
void closelog(void);
```

```
void vsyslog(int priority, const char *format, va_list ap);
```

Feature Test Macro Requirements for glibc (see **feature\_test\_macros(7)**):

```
vsyslog():
```

```
Since glibc 2.19:
```

```
_DEFAULT_SOURCE
```

```
glibc 2.19 and earlier:
```

```
_BSD_SOURCE
```

**DESCRIPTION****openlog()**

**openlog()** opens a connection to the system logger for a program.

The string pointed to by *ident* is prepended to every message, and is typically set to the program name. If *ident* is NULL, the program name is used. (POSIX.1-2008 does not specify the behavior when *ident* is NULL.)

The *option* argument specifies flags which control the operation of **openlog()** and subsequent calls to **syslog()**. The *facility* argument establishes a default to be used if none is specified in subsequent calls to **syslog()**. The values that may be specified for *option* and *facility* are described below.

The use of **openlog()** is optional; it will automatically be called by **syslog()** if necessary, in which case *ident* will default to NULL.

**syslog() and vsyslog()**

**syslog()** generates a log message, which will be distributed by **syslogd(8)**.

The *priority* argument is formed by ORing together a *facility* value and a *level* value (described below). If no *facility* value is ORed into *priority*, then the default value set by **openlog()** is used, or, if there was no preceding **openlog()** call, a default of **LOG\_USER** is employed.

The remaining arguments are a *format*, as in **printf(3)**, and any arguments required by the *format*, except that the two-character sequence **%m** will be replaced by the error message string *strerror(errno)*. The format string need not include a terminating newline character.

The function **vsyslog()** performs the same task as **syslog()** with the difference that it takes a set of arguments which have been obtained using the **stdarg(3)** variable argument list macros.

**closelog()**

**closelog()** closes the file descriptor being used to write to the system logger. The use of **closelog()** is optional.

**Values for option**

The *option* argument to **openlog()** is a bit mask constructed by ORing together any of the following values:

**LOG\_CONS** Write directly to the system console if there is an error while sending to the system logger.

**LOG\_NDELAY** Open the connection immediately (normally, the connection is opened when the first message is logged). This may be useful, for example, if a subsequent **chroot(2)** would make the pathname used internally by the logging facility unreachable.

- LOG\_NOWAIT** Don't wait for child processes that may have been created while logging the message. (The GNU C library does not create a child process, so this option has no effect on Linux.)
- LOG\_ODELAY** The converse of **LOG\_NDELAY**; opening of the connection is delayed until **syslog()** is called. (This is the default, and need not be specified.)
- LOG\_PERROR** (Not in POSIX.1-2001 or POSIX.1-2008.) Also log the message to *stderr*.
- LOG\_PID** Include the caller's PID with each message.

#### Values for *facility*

The *facility* argument is used to specify what type of program is logging the message. This lets the configuration file specify that messages from different facilities will be handled differently.

- LOG\_AUTH** security/authorization messages
- LOG\_AUTHPRIV** security/authorization messages (private)
- LOG\_CRON** clock daemon (**cron** and **at**)
- LOG\_DAEMON** system daemons without separate facility value
- LOG\_FTP** ftp daemon
- LOG\_KERN** kernel messages (these can't be generated from user processes)
- LOG\_LOCAL0** through **LOG\_LOCAL7** reserved for local use
- LOG\_LPR** line printer subsystem
- LOG\_MAIL** mail subsystem
- LOG\_NEWS** USENET news subsystem
- LOG\_SYSLOG** messages generated internally by **syslogd**(8)
- LOG\_USER** (default) generic user-level messages
- LOG\_UUCP** UUCP subsystem

#### Values for *level*

This determines the importance of the message. The levels are, in order of decreasing importance:

- LOG\_EMERG** system is unusable
- LOG\_ALERT** action must be taken immediately
- LOG\_CRIT** critical conditions
- LOG\_ERR** error conditions
- LOG\_WARNING** warning conditions
- LOG\_NOTICE** normal, but significant, condition
- LOG\_INFO** informational message
- LOG\_DEBUG** debug-level message

The function **setlogmask**(3) can be used to restrict logging to specified levels only.

#### ATTRIBUTES

For an explanation of the terms used in this section, see **attributes**(7).

Interface	Attribute	Value
<b>openlog()</b> , <b>closelog()</b>	Thread safety	MT-Safe
<b>syslog()</b> , <b>vsyslog()</b>	Thread safety	MT-Safe env locale

## STANDARDS

The functions **openlog()**, **closelog()**, and **syslog()** (but not **vsyslog()**) are specified in SUSv2, POSIX.1-2001, and POSIX.1-2008.

POSIX.1-2001 specifies only the **LOG\_USER** and **LOG\_LOCAL\*** values for *facility*. However, with the exception of **LOG\_AUTHPRIV** and **LOG\_FTP**, the other *facility* values appear on most UNIX systems.

The **LOG\_PERROR** value for *option* is not specified by POSIX.1-2001 or POSIX.1-2008, but is available in most versions of UNIX.

## NOTES

The argument *ident* in the call of **openlog()** is probably stored as-is. Thus, if the string it points to is changed, **syslog()** may start prepending the changed string, and if the string it points to ceases to exist, the results are undefined. Most portable is to use a string constant.

Never pass a string with user-supplied data as a format, use the following instead:

```
syslog(priority, "%s", string);
```

## SEE ALSO

**journalctl(1)**, **logger(1)**, **setlogmask(3)**, **syslog.conf(5)**, **syslogd(8)**