

NAME

syscalls – Linux system calls

SYNOPSIS

Linux system calls.

DESCRIPTION

The system call is the fundamental interface between an application and the Linux kernel.

System calls and library wrapper functions

System calls are generally not invoked directly, but rather via wrapper functions in glibc (or perhaps some other library). For details of direct invocation of a system call, see **intro(2)**. Often, but not always, the name of the wrapper function is the same as the name of the system call that it invokes. For example, glibc contains a function **truncate()** which invokes the underlying "truncate" system call.

Often the glibc wrapper function is quite thin, doing little work other than copying arguments to the right registers before invoking the system call, and then setting *errno* appropriately after the system call has returned. (These are the same steps that are performed by **syscall(2)**, which can be used to invoke system calls for which no wrapper function is provided.) Note: system calls indicate a failure by returning a negative error number to the caller; when this happens, the wrapper function negates the returned error number (to make it positive), copies it to *errno*, and returns **-1** to the caller of the wrapper.

Sometimes, however, the wrapper function does some extra work before invoking the system call. For example, nowadays there are (for reasons described below) two related system calls, **truncate(2)** and **truncate64(2)**, and the glibc **truncate()** wrapper function checks which of those system calls are provided by the kernel and determines which should be employed.

System call list

Below is a list of those system calls that are common to most platforms. In the list, the *Kernel* column indicates the kernel version for those system calls that were new in Linux 2.2, or have appeared since that kernel version. Note the following points:

- * Where no kernel version is indicated, the system call appeared in kernel 1.0 or earlier.
- * Where a system call is marked "1.2" this means the system call probably appeared in a 1.1.x kernel version, and first appeared in a stable kernel with 1.2. (Development of the 1.2 kernel was initiated from a branch of kernel 1.0.6 via the 1.1.x unstable kernel series.)
- * Where a system call is marked "2.0" this means the system call probably appeared in a 1.3.x kernel version, and first appeared in a stable kernel with 2.0. (Development of the 2.0 kernel was initiated from a branch of kernel 1.2.x, somewhere around 1.2.10, via the 1.3.x unstable kernel series.)
- * Where a system call is marked "2.2" this means the system call probably appeared in a 2.1.x kernel version, and first appeared in a stable kernel with 2.2.0. (Development of the 2.2 kernel was initiated from a branch of kernel 2.0.21 via the 2.1.x unstable kernel series.)
- * Where a system call is marked "2.4" this means the system call probably appeared in a 2.3.x kernel version, and first appeared in a stable kernel with 2.4.0. (Development of the 2.4 kernel was initiated from a branch of kernel 2.2.8 via the 2.3.x unstable kernel series.)
- * Where a system call is marked "2.6" this means the system call probably appeared in a 2.5.x kernel version, and first appeared in a stable kernel with 2.6.0. (Development of kernel 2.6 was initiated from a branch of kernel 2.4.15 via the 2.5.x unstable kernel series.)
- * Starting with kernel 2.6.0, the development model changed, and new system calls may appear in each 2.6.x release. In this case, the exact version number where the system call appeared is shown.
- * In some cases, a system call was added to a stable kernel series after it branched from the previous stable kernel series, and then backported into the earlier stable kernel series. For example some system calls that appeared in 2.6.x were also backported into a 2.4.x release after 2.4.15. When this is so, the version where the system call appeared in both of the major kernel series is listed.

The list of system calls that are available as at kernel 2.6.28 (or in a few cases only on older kernels) is as follows:

System call	Kernel	Notes
_llseek(2)	1.2	
_newselect(2)		
_sysctl(2)		
accept(2)		
accept4(2)	2.6.28	
access(2)		
acct(2)		
add_key(2)	2.6.11	
adjtimex(2)		
afs_syscall(2)		Not implemented
alarm(2)		
alloc_hugepages(2)	2.5.36	Removed in 2.5.44
bdflush(2)		Deprecated (does nothing) since 2.6
bind(2)		
break(2)		Not implemented
brk(2)		
cacheflush(2)	1.2	Not on i386
capget(2)	2.2	
capset(2)	2.2	
chdir(2)		
chmod(2)		
chown(2)		
chown32(2)	2.4	
chroot(2)		
clock_getres(2)	2.6	
clock_gettime(2)	2.6	
clock_nanosleep(2)	2.6	
clock_settime(2)	2.6	
clone(2)		
close(2)		
connect(2)		
creat(2)		
create_module(2)		
delete_module(2)		
dup(2)		
dup2(2)		
dup3(2)	2.6.27	
epoll_create(2)	2.6	
epoll_create1(2)	2.6.27	
epoll_ctl(2)	2.6	
epoll_pwait(2)	2.6.19	
epoll_wait(2)	2.6	
eventfd(2)	2.6.22	
eventfd2(2)	2.6.27	
execve(2)		
exit(2)		
exit_group(2)	2.6	
faccessat(2)	2.6.16	

fcntl64(2)	2.6	
fcntl64_64(2)	2.6	
fallocate(2)	2.6.23	
fchdir(2)		
fchmod(2)		
fchmodat(2)	2.6.16	
fchown(2)		
fchown32(2)	2.4	
fchownat(2)	2.6.16	
fcntl(2)		
fcntl64(2)	2.4	
fdatasync(2)		
fgetxattr(2)	2.6; 2.4.18	
flistxattr(2)	2.6; 2.4.18	
flock(2)	2.0	
fork(2)		
free_hugepages(2)	2.5.36	Removed in 2.5.44
fremovexattr(2)	2.6; 2.4.18	
fsetxattr(2)	2.6; 2.4.18	
fstat(2)		
fstat64(2)	2.4	
fstatat64(2)	2.6.16	
fstatfs(2)		
fstatfs64(2)	2.6	
fsync(2)		
ftime(2)		Not implemented
ftruncate(2)		
ftruncate64(2)	2.4	
futex(2)	2.6	
futimesat(2)	2.6.16	
get_kernel_syms(2)		
get_mempolicy(2)	2.6.6	
get_robust_list(2)	2.6.17	
get_thread_area(2)	2.6	
getcpu(2)	2.6.19	
getcwd(2)	2.2	
getdents(2)	2.0	
getdents64(2)	2.4	
getegid(2)		
getegid32(2)	2.4	
geteuid(2)		
geteuid32(2)	2.4	
getgid(2)		
getgid32(2)	2.4	
getgroups(2)		
getgroups32(2)	2.4	
getitimer(2)		
getpeername(2)		
getpagesize(2)	2.0	Not on i386
getpgid(2)		
getpgrp(2)		
getpid(2)		
getpmsg(2)		Not implemented

getppid(2)		
getpriority(2)		
getresgid(2)	2.2	
getresgid32(2)	2.4	
getresuid(2)	2.2	
getresuid32(2)	2.4	
getrlimit(2)		
getrusage(2)		
getsid(2)	2.0	
getsockname(2)		
getsockopt(2)		
gettid(2)	2.4.11	
gettimeofday(2)		
getuid(2)		
getuid32(2)	2.4	
getxattr(2)	2.6; 2.4.18	
gtty(2)		Not implemented
idle(2)		Not implemented
init_module(2)		
inotify_add_watch(2)	2.6.13	
inotify_init(2)	2.6.13	
inotify_init1(2)	2.6.27	
inotify_rm_watch(2)	2.6.13	
io_cancel(2)	2.6	
io_destroy(2)	2.6	
io_getevents(2)	2.6	
io_setup(2)	2.6	
io_submit(2)	2.6	
ioctl(2)		
ioperm(2)		
iopl(2)		
ioprio_get(2)	2.6.13	
ioprio_set(2)	2.6.13	
ipc(2)		
kexec_load(2)	2.6.7	
keyctl(2)	2.6.11	
kill(2)		
lchown(2)	2.2	
lchown32(2)	2.4	
lgetxattr(2)	2.6; 2.4.18	
link(2)		
linkat(2)	2.6.16	
listen(2)		
listxattr(2)	2.6; 2.4.18	
llistxattr(2)	2.6; 2.4.18	
lock(2)		Not implemented
lookup_dcookie(2)	2.6	
lremovexattr(2)	2.6; 2.4.18	
lseek(2)		
lsetxattr(2)	2.6; 2.4.18	
lstat(2)		
lstat64(2)	2.4	
madvise(2)	2.4	

madvise1(2)	2.4	
mbind(2)	2.6.6	
migrate_pages(2)	2.6.16	
mincore(2)	2.4	
mkdir(2)		
mkdirat(2)	2.6.16	
mknod(2)		
mknodat(2)	2.6.16	
mlock(2)		
mlockall(2)		
mmap(2)		
mmap2(2)	2.4	
modify_ldt(2)		
mount(2)		
move_pages(2)	2.6.18	
mprotect(2)		
mpx(2)		Not implemented
mq_getsetattr(2)	2.6.6	
mq_notify(2)	2.6.6	
mq_open(2)	2.6.6	
mq_timedreceive(2)	2.6.6	
mq_timedsend(2)	2.6.6	
mq_unlink(2)		
mremap(2)	2.0	
msgctl(2)		
msgget(2)		
msgrcv(2)		
msgsnd(2)		
msync(2)	2.0	
munlock(2)		
munlockall(2)		
munmap(2)		
nanosleep(2)	2.0	
nfsservctl(2)	2.2	
nice(2)		
oldfstat(2)		
oldlstat(2)		
oldolduname(2)		
oldstat(2)		
olduname(2)		
open(2)		
openat(2)	2.6.16	
pause(2)		
pciconfig_iobase(2)	2.2.15; 2.4	Not on i386
pciconfig_read(2)	2.0.26; 2.2	Not on i386
pciconfig_write(2)	2.0.26; 2.2	Not on i386
personality(2)	1.2	
phys(2)		Not implemented
pipe(2)		
pipe2(2)	2.6.27	
pivot_root(2)	2.4	
poll(2)	2.2	
ppoll(2)	2.6.16	

prctl(2)	2.2	
pread64(2)		Added as "pread" in 2.2; renamed "pread64" in 2.6
preadv(2)	2.6.30	
prof(2)		Not implemented
profil(2)		Not implemented
pselect6(2)	2.6.16	
ptrace(2)		
putpmsg(2)		Not implemented
pwrite64(2)		Added as "pwrite" in 2.2; renamed "pwrite64" in 2.6
pwritev(2)	2.6.30	
query_module(2)	2.2	
quotactl(2)		
read(2)		
readahead(2)	2.4.13	
readdir(2)		
readlink(2)		
readlinkat(2)	2.6.16	
readv(2)	2.0	
reboot(2)		
recv(2)		
recvfrom(2)		
recvmsg(2)		
remap_file_pages(2)	2.6	
removexattr(2)	2.6; 2.4.18	
rename(2)		
renameat(2)	2.6.16	
request_key(2)	2.6.11	
restart_syscall(2)	2.6	
rmdir(2)		
rt_sigaction(2)	2.2	
rt_sigpending(2)	2.2	
rt_sigprocmask(2)	2.2	
rt_sigqueueinfo(2)	2.2	
rt_sigreturn(2)	2.2	
rt_sigsuspend(2)	2.2	
rt_sigtimedwait(2)	2.2	
sched_get_priority_max(2)	2.0	
sched_get_priority_min(2)	2.0	
sched_getaffinity(2)	2.6	
sched_getparam(2)	2.0	
sched_getscheduler(2)	2.0	
sched_rr_get_interval(2)	2.0	
sched_setaffinity(2)	2.6	
sched_setparam(2)	2.0	
sched_setscheduler(2)	2.0	
sched_yield(2)	2.0	
security(2)		Not implemented
select(2)		
semctl(2)		
semget(2)		
semop(2)		

semtimedop(2)	2.6; 2.4.22	
send(2)		
sendfile(2)	2.2	
sendfile64(2)	2.6; 2.4.19	
sendmsg(2)		
sendto(2)		
set_mempolicy(2)	2.6.6	
set_robust_list(2)	2.6.17	
set_thread_area(2)	2.6	
set_tid_address(2)	2.6	
set_zone_reclaim(2)	2.6.13	Removed in 2.6.16 (was never available to userspace)
setdomainname(2)		
setfsuid(2)	1.2	
setfsuid32(2)	2.4	
setfsuid(2)	1.2	
setfsuid32(2)	2.4	
setgid(2)		
setgid32(2)	2.4	
setgroups(2)		
setgroups32(2)	2.4	
sethostname(2)		
setitimer(2)		
setpgid(2)		
setpriority(2)		
setregid(2)		
setregid32(2)	2.4	
setresgid(2)	2.2	
setresgid32(2)	2.4	
setresuid(2)	2.2	
setresuid32(2)	2.4	
setreuid(2)		
setreuid32(2)	2.4	
setrlimit(2)		
setsid(2)		
setsockopt(2)		
settimeofday(2)		
setuid(2)		
setuid32(2)	2.4	
setup(2)		Removed in 2.2
setxattr(2)	2.6; 2.4.18	
sgetmask(2)		
shmat(2)		
shmctl(2)		
shmdt(2)		
shmget(2)		
shutdown(2)		
sigaction(2)		
sigaltstack(2)	2.2	
signal(2)		
signalfd(2)	2.6.22	
signalfd4(2)	2.6.27	
sigpending(2)		

sigprocmask(2)		
sigreturn(2)		
sigsuspend(2)		
socket(2)		
socketcall(2)		
socketpair(2)		
splice(2)	2.6.17	
spu_create(2)	2.6.16	PowerPC only
spu_run(2)	2.6.16	PowerPC only
ssetmask(2)		
stat(2)		
stat64(2)	2.4	
statfs(2)		
statfs64(2)	2.6	
stime(2)		
stty(2)		Not implemented
swapoff(2)		
swapon(2)		
symlink(2)		
symlinkat(2)	2.6.16	
sync(2)		
sync_file_range(2)	2.6.17	
sysfs(2)	1.2	
sysinfo(2)		
syslog(2)		
tee(2)	2.6.17	
tgkill(2)	2.6	
time(2)		
timer_create(2)	2.6	
timer_delete(2)	2.6	
timer_getoverrun(2)	2.6	
timer_gettime(2)	2.6	
timer_settime(2)	2.6	
timerfd_create(2)	2.6.25	
timerfd_gettime(2)	2.6.25	
timerfd_settime(2)	2.6.25	
times(2)		
tkill(2)	2.6; 2.4.22	
truncate(2)		
truncate64(2)	2.4	
tuxcall(2)		Not implemented
ugetrlimit(2)	2.4	
ulimit(2)		Not implemented
umask(2)		
umount(2)		
umount2(2)	2.2	
uname(2)		
unlink(2)		
unlinkat(2)	2.6.16	
unshare(2)	2.6.16	
uselib(2)		
ustat(2)		
utime(2)		

utimensat(2)	2.6.22	
utimes(2)	2.6	
vfork(2)		
vhangup(2)		
vm86old(2)		
vmsplice(2)	2.6.17	
vserver(2)		Not implemented
wait4(2)		
waitid(2)	2.6.10	
waitpid(2)		
write(2)		
writew(2)	2.0	

On many platforms, including i386, socket calls are all multiplexed (via glibc wrapper functions) through **socketcall(2)** and similarly System V IPC calls are multiplexed through **ipc(2)**.

NOTES

Note the following points:

- * Although slots are reserved for them in the system call table, the following system calls are not implemented in the standard kernel: **afs_syscall(2)**, **break(2)**, **ftime(2)**, **getpmsg(2)**, **gty(2)**, **idle(2)**, **lock(2)**, **madvisel(2)**, **mpx(2)**, **phys(2)**, **prof(2)**, **profil(2)**, **putpmsg(2)**, **security(2)**, **stty(2)**, **tuxcall(2)**, **ulimit(2)**, and **vserver(2)** (see also **unimplemented(2)**). However, **ftime(3)**, **profil(3)** and **ulimit(3)** exist as library routines. The slot for **phys(2)** is in use since kernel 2.1.116 for **umount(2)**; **phys(2)** will never be implemented.
- * The **getpmsg(2)** and **putpmsg(2)** calls are for kernels patched to support STREAMS, and may never be in the standard kernel.

Roughly speaking, the code belonging to the system call with number `__NR_xxx` defined in `/usr/include/asm/unistd.h` can be found in the kernel source in the routine `sys_xxx()`. (The dispatch table for i386 can be found in `/usr/src/linux/arch/i386/kernel/entry.S`.) There are many exceptions, however, mostly because older system calls were superseded by newer ones, and this has been treated somewhat unsystematically. On platforms with proprietary OS emulation, such as parisc, sparc, sparc64 and alpha, there are many additional system calls; mips64 also contains a full set of 32-bit system calls.

Over time, changes to the interfaces of some system calls have been necessary. One reason for such changes was the need to increase the size of structures or scalar values passed to the system call. Because of these changes, there are now various groups of related system calls (e.g., **truncate(2)** and **truncate64(2)**) which perform similar tasks, but which vary in details such as the size of their arguments. (As noted earlier, applications are generally unaware of this: the glibc wrapper functions do some work to ensure that the right system call is invoked, and that ABI compatibility is preserved for old binaries.) Examples of systems calls that exist in multiple versions are the following:

- * By now there are three different versions of **stat(2)**: `sys_stat()` (slot `__NR_oldstat`), `sys_newstat()` (slot `__NR_stat`), and `sys_stat64()` (slot `__NR_stat64`), with the last being the most current. A similar story applies for **lstat(2)** and **fstat(2)**.
- * Similarly, the defines `__NR_oldolduname`, `__NR_olduname`, and `__NR_uname` refer to the routines `sys_olduname()`, `sys_uname()` and `sys_newuname()`.
- * In Linux 2.0, a new version of **vm86(2)** appeared, with the old and the new kernel routines being named `sys_vm86old()` and `sys_vm86()`.
- * In Linux 2.4, a new version of **getrlimit(2)** appeared, with the old and the new kernel routines being named `sys_old_getrlimit()` (slot `__NR_getrlimit`) and `sys_getrlimit()` (slot `__NR_ugetrlimit`).
- * Linux 2.4 increased the size of user and group IDs from 16 to 32 bits. To support this change, a range of system calls were added (e.g., **chown32(2)**, **getuid32(2)**, **getgroups32(2)**, **setresuid32(2)**), superseding earlier calls of the same name without the "32" suffix.

- * Linux 2.4 added support for applications on 32-bit architectures to access large files (i.e., files for which the sizes and file offsets can't be represented in 32 bits.) To support this change, replacements were required for system calls that deal with file offsets and sizes. Thus the following system calls were added: **fcntl64(2)**, **ftruncate64(2)**, **getdents64(2)**, **stat64(2)**, **statfs64(2)**, and their analogs that work with file descriptors or symbolic links. These system calls supersede the older system calls which, except in the case of the "stat" calls, have the same name without the "64" suffix.

On newer platforms that only have 64-bit file access and 32-bit uids (e.g., alpha, ia64, s390x) there are no *64 or *32 calls. Where the *64 and *32 calls exist, the other versions are obsolete.

- * The *rt_sig** calls were added in kernel 2.2 to support the addition of real-time signals (see **signal(7)**). These system calls supersede the older system calls of the same name without the "rt_" prefix.
- * The **select(2)** and **mmap(2)** system calls use five or more arguments, which caused problems the way argument passing on the i386 used to be set up. Thus, while other architectures have *sys_select()* and *sys_mmap()* corresponding to *__NR_select* and *__NR_mmap*, on i386 one finds *old_select()* and *old_mmap()* (routines that use a pointer to a argument block) instead. These days passing five arguments is not a problem any more, and there is a *__NR_newselect* that corresponds directly to *sys_select()* and similarly *__NR_mmap2*.

SEE ALSO

syscall(2), **unimplemented(2)**, **libc(7)**

COLOPHON

This page is part of release 3.22 of the Linux *man-pages* project. A description of the project, and information about reporting bugs, can be found at <http://www.kernel.org/doc/man-pages/>.