Linux System Programming: System Calls & API Reference

open()

Opens a file and returns a file descriptor.

Syntax: int open(const char *pathname, int flags, mode_t mode);

- flags: O_RDONLY, O_WRONLY, O_RDWR, O_CREAT, O_TRUNC, O_APPEND

- mode: file permissions (used with O_CREAT)

Returns: non-negative file descriptor on success, -1 on error.

read()

Reads data from a file descriptor into a buffer.

Syntax: ssize_t read(int fd, void *buf, size_t count);

- fd: file descriptor to read from

- buf: buffer to store read data

- count: number of bytes to read

Returns: number of bytes read, 0 on EOF, -1 on error.

write()

Writes data from a buffer to a file descriptor.

Syntax: ssize_t write(int fd, const void *buf, size_t count);

- fd: file descriptor to write to

- buf: data to write

- count: number of bytes

Returns: number of bytes written or -1 on error.

lseek()

Moves the file offset.

Syntax: off_t lseek(int fd, off_t offset, int whence);

- whence: SEEK_SET, SEEK_CUR, SEEK_END

Returns: new offset or -1 on error.

close()

Closes a file descriptor.

Syntax: int close(int fd);

Releases the file descriptor.

Returns: 0 on success, -1 on error.

fork()

Creates a new process by duplicating the calling process.

Syntax: pid_t fork();

Returns: 0 to child, child's PID to parent, -1 on error.

wait() / waitpid()

wait(): Waits for any child process to terminate.

Syntax: pid_t wait(int *status);

waitpid(): Waits for a specific child process.

Syntax: pid_t waitpid(pid_t pid, int *status, int options);

dup() / dup2()

Duplicates file descriptors.

dup(fd): returns lowest unused FD

dup2(oldfd, newfd): forces duplication into newfd (closing it first).

stat() / lstat()

Retrieve file metadata.

stat(): follows symlinks, lstat(): does not follow.

Returns data in struct stat.

chmod() / umask()

chmod(): changes file permissions.

umask(): sets default permission mask for newly created files.

utime()

Sets access and modification times of a file.

Syntax: int utime(const char *filename, const struct utimbuf *times);

opendir(), readdir(), closedir()

Used for directory handling.

- opendir(): open directory stream

- readdir(): read entries

- closedir(): close stream

unlink()

Deletes a name from the filesystem.

Syntax: int unlink(const char *pathname);

File is deleted only when no file descriptors refer to it.

signal() / sigaction()

Used for signal handling.

signal(sig, handler): sets a signal handler.

sigaction(): more robust and reliable version for setting handlers.