|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

**Отчёт по лабораторной работе №6 по курсу**

**«Операционные системы»**

**Тема**  Системный вызов open()

**Студент**  Якуба Д.В.

**Группа**  ИУ7-63Б

**Оценка (баллы)**

**Преподаватель**  Рязанова Н.Ю.

*Москва, 2021 г*

# Необходимые структуры

Версия ядра: v5.13-rc3

struct [**filename**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/filename) {

const char \*name; */\* pointer to actual string \*/*

const [**\_\_user**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/__user) char \***[uptr](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/uptr)**; */\* original userland pointer \*/*

int [**refcnt**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/refcnt);

struct [**audit\_names**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/audit_names) \***[aname](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/aname)**;

const char [**iname**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/iname)[];

};

struct [**open\_flags**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/open_flags) {

int [**open\_flag**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/open_flag);

[**umode\_t**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/umode_t) mode;

int [**acc\_mode**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/acc_mode);

int [**intent**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/intent);

int [**lookup\_flags**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/lookup_flags);

};

#define [**EMBEDDED\_LEVELS**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/EMBEDDED_LEVELS) 2

struct [**nameidata**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/nameidata) {

struct [**path**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/path) [**path**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/path);

struct [**qstr**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/qstr) [**last**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/last);

struct [**path**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/path) root;

struct [**inode**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/inode) \***[inode](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/inode)**; */\* path.dentry.d\_inode \*/*

unsigned int flags;

unsigned [**seq**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/seq), [**m\_seq**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/m_seq), [**r\_seq**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/r_seq);

int [**last\_type**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/last_type);

unsigned [**depth**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/depth);

int [**total\_link\_count**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/total_link_count);

struct [**saved**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/saved) {

struct [**path**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/path) [**link**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/link);

struct [**delayed\_call**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/delayed_call) [**done**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/done);

const char \*name;

unsigned [**seq**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/seq);

} \*[**stack**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/stack), [**internal**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/internal)[[**EMBEDDED\_LEVELS**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/EMBEDDED_LEVELS)];

struct [**filename**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/filename) \*name;

struct [**nameidata**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/nameidata) \*[**saved**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/saved);

unsigned [**root\_seq**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/root_seq);

int [**dfd**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/dfd);

[**kuid\_t**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/kuid_t) [**dir\_uid**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/dir_uid);

[**umode\_t**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/umode_t) [**dir\_mode**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/dir_mode);

} [**\_\_randomize\_layout**](https://elixir.bootlin.com/linux/v5.13-rc3/C/ident/__randomize_layout);

struct path {

struct vfsmount \*mnt;

struct dentry \*dentry;

} \_\_randomize\_layout;

Флаги, используемые с вызовом open()

**O\_CREAT –** если файл не существует, то он будет создан

**O\_EXCL –** если используется совместно с O\_CREAT, то при наличии уже созданного файла вызов завершится ошибкой

**O\_NOCTTY –** если файл указывает на терминальное устройство, то оно не станет терминалом управления процесса, даже при его отсутствии

**O\_TRUNC –** если файл уже существует, он является обычным файлом и заданный режим позволяет записывать в этот файл, то его длина будет урезана до нуля

**O\_APPEND –** файл открывается в режиме добавления, перед каждой операцией записи файловый указатель будет устанавливаться в конец файла

**O\_NONBLOCK, O\_NDELAY –** файл открывается, по возможности, в режиме non-blocking, то есть никакие последующие операции над дескриптором файла не заставляют в дальнейшем вызывающий процесс ждать

**O\_SYNC –** файл открывается в режиме синхронного ввода-вывода, то есть все операции записи для соответствующего дескриптора файла блокируют вызывающий процесс до тех пор, пока данные не будут физически записаны

**O\_NOFOLLOW –** если файл является символической ссылкой, то open вернёт ошибку

**O\_DIRECTORY –** если файл не является каталогом, то open вернёт ошибку

**O\_LARGEFILE –** позволяет открывать файлы, размер которых не может быть представлен типом off\_t (long)

**O\_DSYNC –** операции записи в файл будут завершены в соответствии с требованиями целостности данных синхронизированного завершения ввода-вывода

**O\_NOATIME –** запрет на обновление времени последнего доступа к файлу при его чтении

**O\_TMPFILE –** при наличии данного флага создаётся неименованный временный обычный файл

**O\_CLOEXEC –** включает флаг close-on-exec для нового файлового дескриптора, указание этого флага позволяет программе избегать дополнительных операций fcntl F\_SETFD для установки флага FD\_CLOEXEC





 





