Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем

Киреев Илья Владимирович

# Исследование размера максимальной длины имени файла в Linux

Учебная практика

Научный руководитель:
старший преподаватель кафедры
системного программирования Я.А. Кириленко

Консультант:
генеральный директор Etersoft В.А. Липатов

Санкт-Петербург
2021

# Оглавление

# Введение

Максимальная длина имени файла, как и множество других его параметров, определяются конкретной файловой системой, в которой этот файл находится. Linux поддерживает множество различных файловых систем, поэтому для исключения проблем с длиной имен файлов, которые могут возникнуть при переносе данных с Windows, хотелось бы добиться наилучшей возможной совместимости этих операционных систем, приведя этот параметр к единому значению.

Мотивацией заняться этой задачей послужила популяризация свободного программного обеспечения, в частности, переход организаций с Windows на Linux. В настоящее время все больше компаний и государственных структур переходят на использование в своей работе свободного программного обеспечения (СПО) — программного обеспечения, которое распространяется с открытым исходным кодом на условиях одной их свободных лицензией[1], передающей пользователю от автора права на следующие свободы действий:

- ПО можно использовать в любых целях («нулевая свобода»).
- ПО можно изучать и адаптировать под свои задачи («первая свобода»).
- Копии ПО можно свободно распространять («вторая свобода»).
- Можно свободно улучшать ПО и публиковать внесенные изменения («третья свобода»)

Такой подход позволяет существенно снизить затраты на разработку: необходимо оплатить создание уникального решения, а не каждый установленный экземпляр продукта. Нет нужды каждый раз разрабатывать основные составляющие программного обеспечения, тратя на это ресурсы, можно адаптировать имеющееся ПО под свои задачи. Также СПО более безопасно, в силу открытости исходного кода. Проприе-

[1] `https://www.gnu.org/licenses/licenses.en.html`

тарные продукты от известных производителей могут содержать недокументированную функциональность и из–за необходимости в информационной безопасности использовать их в работе не всегда возможно.

Etersoft[2] — российская компания, занимающаяся разработкой программных решений на основе СПО. Etersoft популяризует свободное программное обеспечение и разрабатывает решения для организаций, позволяющие работать с Linux и свободными программами. Одним из направлений работы является создание инструментов, для перехода организаций с Windows на Linux и для этого необходима наилучшая возможная совместимость этих систем, для переноса данных.

В Windows для именования файлов используется кодировка UTF–16[3], где каждый символ кодируется двумя байтами, а максимальная длина имени файла равняется 255 символов (510 байтов). В Linux же для задания имени файла применяется кодировка UTF–8, при этом максимальное значение этого параметра задается в glibc[4], и равняется 255 байтам. К данной константе обращаются программы, использующие этот параметр, а реальное значение определяется используемой файловой системой. Получаем ограничение в 127 символов, при использовании русского алфавита. В связи с этим возникает проблема, что длинные имена файлов (от 127 до 255 символов) не умещаются в это ограничение Linux.

[2] http://etersoft.com

[3] https://en.wikipedia.org/wiki/UTF-16

[4] https://www.gnu.org/software/libc/

## Постановка задачи

Целью моей работы является приведение максимальной длины имени файла в обозначенных файловых системах к единому значению. Для достижения этой цели были поставлены следующие задачи.

1. Протестировать выбранные файловые системы на пример максимального значения длины имени файла и сравнить заявленные, тестовые и реально доступные значения.

2. Написать патч, для приведения значения максимальной длины имени файла в обозначенных системах к единому значению.

3. Апробировать результаты.

# 1. Обзор

## 1.1. Выбор файловых систем для анализа

Стандартной файловой системой для Linux на момент тестирования является EXT4. Также в некоторых дистрибутивах встречаются относительно новые Btrfs и ZFS, которые не так давно получили статус стабильных. Для переносных же носителей наиболее популярны FAT32, exFAT и NTFS. Поэтому было принято решение выбрать для анализа и исправления перечисленные файловые системы.

## 1.2. Эмуляция раздела диска с необходимой файловой системой

Для проведения тестирования и впоследствии исправления как можно большего числа файловых систем, возникла необходимость в их эмуляции на машине. Первым этапом, командой truncate [1], создавался файл, далее в этом файле, с помощью утилиты mkfs [2] создавалась файловая система необходимого типа. Этот файл монтировался с помощью инструмента mount [5] в пустой каталог, в которой в дальнейшем и проводилось тестирование. После выполнения тестов, утилитой umount [8], файловая система размонтировалась, файл с каталогом удалялись.

## 1.3. Statfs

Для получения статистических данных о файловой системе, в частности типа и максимальной длины имени файла, была написана тестовая программа с применением библиотечной функции statfs[7]. Функция позволяет по переданному пути до файла получить информацию о файловой системе, в которой он находится. Была добавлена проверка операционной системы, где запускается приложение, поскольку реализация для macOS не содержит поля с предельной длиной имени, также добавлено приведение численного значения типа файловой системы к читаемому эквиваленту.

# 2. Тестирование и анализ файловых систем

В этой главе приводится тестирование файловых систем и исследование кода ядра, с целью изучения максимальной длины имени файла, которую:

- Возвращает библиотечная функция statfs [7].
- Реально возможно создать в данной файловой системе.
- Заявляет производитель.

Работа выполнялась на дистрибутиве Ubuntu 20.10 (Groovy Gorilla)[5] с версией ядра 5.8.0. Программа и скрипт для проверки корректности полученных значений доступны в репозитории[6]. Были получены следующие результаты:

| | Заявлено | Statfs | Реальный максимум латиницы | Реальный максимум кириллицы |
|---|---|---|---|---|
| FAT32 LFN | 255 символов UTF–16 | 1530 | 255 символов | 127 символов |
| exFAT | 255 символов UTF–16 | 1530 | 255 символов | 255 символов |
| ext4 | 255 байт | 255 | 255 символов | 127 символов |
| btrfs | 255 байт | 255 | 255 символов | 127 символов |
| NTFS | 255 символов UTF–16 | 255 | 0 символов | 0 символов |
| NTFS FUSE | 255 символов UTF–16 | 255 | 255 символов | 255 символов |
| ZFS | 255 символов UTF–8 | 255 | 255 символов | 127 символов |

## 2.1. FAT32

В Windows 95 для файловой системы FAT[7] стало доступно расширение VFAT, которое добавило поддержку длинных (до 255 символов) имен в кодировке UTF–16. Тестирование проводилось с применением этого расширения. Поле f_namelen структуры statfs для данной системы возвращает значение 1530. В ядре это число вычисляется как FAT_LFN_LEN умножить на NLS_MAX_CHARSET_SIZE, где первая константа равняется 255, а вторая 6. Хотя с ноября 2003 года пятый и шестой байты в UTF–8 уже не используются при кодировании, для

[5] `https://releases.ubuntu.com/20.10/`

[6] `https://github.com/foiki/longFileName`

[7] `https://en.wikipedia.org/wiki/File_Allocation_Table`

обеспечения совместимости с UTF–16, эта константа осталась без изменений. Отсюда получаем, что возвращаемое значение соответствует максимальному количеству байт в имени файла.

Для данной файловой системы реальное доступное максимальное значение имени файла равняется 255 символов для латинского алфавита и 127 для кириллицы, что не соответствует заявленному значению.

## 2.2. exFAT

Аналогично FAT32, для exFAT заявляется[8] предельная длина имени файла равна 255 символов UTF–16. Данная файловая система поддерживается в ядре напрямую начиная с версии ядра 5.4, до этого создание производилось с помощью FUSE[9]. Пакет exfat–fuse был удален, а exfatprogs, содержащий утилиту для создания и монтирования рассматриваемой системы, установлен отдельно.

После выполнения программы, значение поля f_namelen равняется 1530, в ядре это значение вычисляется аналогично FAT с расширением VFAT с соответствующим расположением и значениями констант. Полученное значение равняется максимальному количеству байт, используемых для имени файла.

Для данной файловой системы по факту возможно создать имя файла максимум в 255 символов латинского алфавита, такое же значение получилось и для русских символов, что соответствует заявленному значению.

## 2.3. ext4

Для ext4 заявляется[10] предельная длина имени файла в 255 байт и тестовая программа возвращает это же значение. В ядре это поле приравнивается константе EXT4_NAME_LEN равной 255.

Для данной файловой системы реально реально возможно задать

[8] https://docs.microsoft.com/en-us/windows/win32/fileio/exfat-specification

[9] https://en.wikipedia.org/wiki/Filesystem_in_Userspace

[10] https://en.wikipedia.org/wiki/Ext4

имя максимум в 255 латинских символов и в 127 русских, поскольку в Linux для кодирования имени файла применяется UTF–8, где кириллица кодируется двумя байтами на символ, то это равно заявленному значению.

## 2.4. btrfs

Для Btrfs производителем заявляется[11] предельное значение в 255 символов ASCII таблицы для задания имени файла. Для создания данной файловой системы установили пакет btrfs–progs. Поле f_namelen структуры statfs после выполнения тестовой программы равняется 255, в ядре оно присваивается равным константе BTRFS_NAME_LEN, которая равняется 255.

В выбранной системе получилось создать файл максимум с 255 латинскими символами в названии файла и со 127 русскими, а это равно заявленному значению.

## 2.5. NTFS

На момент проведения исследования файловая система NTFS поддерживается в ядре только в режиме чтения, полноценная реализация доступна через FUSE, поэтому тестирование проводилось для обоих вариантов. Для NTFS максимальная длина имени файла должна[12] равняться 255 символам UTF–16.

### 2.5.1. NTFS FUSE

Для создания и монтирования NTFS FUSE установили пакет ntfs–3g. Рассматриваемая файловая система может быть примонтирована только в виде блочного устройства, воспользовались утилитой losetup [4] для поиска свободного петлевого[13] устройства и привязки к нему наше-

[11] https://en.wikipedia.org/wiki/Btrfs

[12] https://en.wikipedia.org/wiki/NTFS

[13] https://en.wikipedia.org/wiki/Loop_device

го файла с NTFS. При запуске тестовой программы получили значение f_namelen равное 255.

При реальном тестировании удалось создать файл максимум с 255 символами латиницы или кириллицы в имени, что соответствует заявленному значению.

### 2.5.2. NTFS read–only

Утилита для создания NTFS включена в пакет ntfs–3g и не поставляется отдельно, установив его и запустив программу для теста было замечено, что реализация через FUSE перекрывает реализацию в ядре и не позволяет примонтировать систему с исполнением в ядре. Поэтому после создания файла с NTFS, пакет был удален и монтирование происходило через нативную реализацию. Получили следующее значение f_namelen: 255.

При попытке создания файлов в такой системе была получена ошибка, что соответствует ожидаемому результату, так как на данный момент поддержка NTFS в ядре в режиме записи находится в разработке.

## 2.6. ZFS

После того, как компания Oracle выкупила SUN Microsystems, которая занималась разработкой и закрыла исходный код данной файловой системы, в 2013 году была выпущена OpenZFS, которая распространяется с открытым исходным кодом и сейчас доступна в некоторых дистрибутивах Linux, поэтому в данной работе будет рассматриваться именно OpenZFS и для упрощения именования в работе будет называться просто ZFS.

Для данной файловой системы заявляется[14] максимальная длина имени файла в 255 символов ASCII таблицы. Для работы с ней установили пакет утилит zfsutils–linux. ZFS может расположена на нескольких разделах, для этого предварительно создается пул и к нему уже

[14] https://en.wikipedia.org/wiki/OpenZFS

подключаются разделы. Выбрали свободное петлевое устройство, привязали к нему файл, для задания непустого размера, создали пул на выбранном петлевом устройстве и задали точку монтирования в предварительно созданную пустую директорию для удобства тестирования.

При запуске тестовой программы получили значение f_namelen равное 255. В коде модуля оно присваивается равным MAXNAMELEN - 1, где константа равняется 256. В ZFS получилось создать файл максимум с 255 латинскими символами в названии файла и со 127 русскими, а это равно заявленному значению.

# 3. Увеличение максимальной длины имени файла

Все файловые системы являются модулями для ядра, что значительно облегчает их сборку: можно собрать только данный модуль, без необходимости полной сборки всего ядра. Некоторые файловые системы используются дистрибутивом постоянно, например в качестве основной или в загрузчике системы, поэтому они предустановлены в ядро и в таких случаях приходилось собирать все ядро для внесения изменений.

В качестве нового максимального значения максимальной длины имени файла было решено взять 1023 байта, что позволит также существенно расширить возможности для использования в именах файлов иероглифов, которые кодируются 4 байтами и при стандартном ограничении 255 байт дают максимум всего в 63 символа.

Для всех утилит, работающих с файловыми системам, в ядре существует общая константа для максимальной длины имени файла, к которой они обращаются, и она была приведена к выбранному значению.

## 3.1. FAT32

В дистрибутиве Ubuntu, на котором проводились тестирования, FAT32 является файловой системой загрузчика, поэтому для внесения изменений и последующей отладки корректности изменений проводилась полная сборка и установка ядра. Из директории FAT может собираться сразу 3 файловых системы, в зависимости от конфигурации ядра: FAT16, FAT32 и MSDOS—FAT. Если не указан определенный тип, то файловая система для сборки выбирается в зависимости от размера файла, поэтому в процессе тестирования при изменении размера тестового файла тип автоматически менялся с FAT32 на FAT16, что приводило к некорректным результатам.

Максимальную длину имени файла для FAT32 с VFAT ограничивает одна константа, исправив ее и собрав ядро были полученны необходи-

мые результаты.

## 3.2. exFAT

Файловая система exFAT загружается в ядро в виде модуля, что значительно упрощает процесс тестирования и отладки. В исходном коде у exFAT существует две равные константы, которые отвечают за максимальную длину имени файла, исправив их и установив исправленный модуль в процессе тестирования возникла ошибка: длины имени в 255, 512 и 768 символов были недоступны для создания.

Для отладки процесса создания файлов были использованы утилиты dmesg [3] и strace [6]. Было обнаружено, что в структуре, отвечающей за имена файлов длина имени задавалась типом char, что приводило к обнулению чисел кратных 255 и некорректной работе с именами большей длины. Данный тип был исправлен на short. Такие изменения не привели к возникновению ошибок в работе системы, поскольку во всех функциях параметр длины имени задавался как int и только при записи в структуру преобразовывался в char.

## 3.3. ext4

Хранение данных в файловой системе ext4 устроено иначе, чем в тех, которые были рассмотрены ранее. В самых ранних версиях под хранение длины имени файла было отведено два байта, что могло обеспечить использование имен файлов до 512 байт длиной, но начиная с версии ext2 второй байт был заинствован под хранение флагов, а так как файловая система ext4 совместима с версиями ext2 и ext3, исправление данного параметра привело бы к потере совместимости, чего допускать не хотелось.

Также, поскольку, ext4 является основной в выбранном дистрибутиве ее исправление требует предварительной смены ее на другую файловую систему, например на btrfs или zfs, поэтому было принято решение на данном этапе не модифицировать ext4.

## 3.4. btrfs

Для btrfs значение максимальной длины имени файла ограничивается одной константой и файловая система загружается в виде модуля, что позволило быстро исправить это значение и протестировать файловую систему. Данные изменения не вызвали никаких ошибок в работе, поскольку btrfs изначально поддерживает имена файлов большей длины, но они были ограничены общим для Linux значением.

## 3.5. NTFS и NTFS FUSE

Ntfs аналогично устанавливается в ядро в виде модуля и исправление константы, ограничивающей максимальную длину имени файлов не вызывает затруднений, но поскольку данная файловая система все еще поддерживается в ядре только в режиме read–only, то проверить исправления с помощью создания файлов не представляется возможным.

Так как модуль FUSE изначально подразумевает создание файловых систем из пользовательского пространства, то разумеется в ядре не содержится ntfs для данного модуля, а загружается отдельным пакетом. Данный модуль был протестирован с целью сравнения с ntfs доступной в ядре.

## 3.6. ZFS

Zfs на данный момент не имеет реализации в ядре, она предустановлена в виде модуля в ряд дистрибутивов, в том числе и Ubuntu.

С официального сайта[15] были загружены исходники данной операционной системы, исправлена константа, отвечающая за максимальную длину имени, модуль был собран и установлен. Изменения также не вызвали ошибок в работе системы.

[15] `https://zfsonlinux.org`

# 4. Апробация результатов

В этой главе приводится апробация внесенных изменений. Тестирование проводилось на актуальной на момент проведения тестирования версии Ubuntu 21.04 (Hirsute Hippo) с версией ядра 5.11.0. Были получены следующие результаты:

| | Заявлено производителем | Statfs | Реальный максимум латиницы | Реальный максимум кириллицы |
|---|---|---|---|---|
| FAT32 LFN | 255 символов UTF–16 | 6138 | 1023 символа | 511 символов |
| exFAT | 255 символов UTF–16 | 6138 | 1023 символа | 511 символов |
| ext4 | 255 байт | 255 | 255 символов | 127 символов |
| btrfs | 255 байт | 1023 | 1023 символа | 511 символов |
| NTFS | 255 символов UTF–16 | 1023 | 0 символов | 0 символов |
| ZFS | 255 символов UTF–16 | 1023 | 1023 символов | 511 символов |

# Заключение

В ходе работы были достигнуты следующие результаты, которые доступны в открытом репозитории[16].

1. Протестированы выбранные файловые системы на пример максимального значения длины имени файла и сравнены заявленные, тестовые и реально доступные значения данного параметра.

2. Написан патч, для приведения значения максимальной длины имени файла в обозначенных системах к единому значению.

3. Апробированы результаты.

[16] `https://github.com/foiki/longFileName`

# Список литературы


[1] Brady Padraig. — truncate(1) - Linux manual page, 2020.

[2] David Engel Fred N. van Kempen Ron Sommeling. — mkfs(8) - Linux manual page, 2011.

[3] Karel Zak Theodore Ts'o. — dmesg(1) - Linux manual page, 2012.

[4] Karel Zak Theodore Ts'o. — losetup(8) - Linux manual page, 2015.

[5] Zak Karel. — mount(8) - Linux manual page, 2015.

[6] stace(1) - Linux manual page, 2020.

[7] statfs(2) - Linux manual page, 2020.

[8] umount(8) - Linux manual page, 2014.