File systems (Русский)

(Redirected from Файловые системы)

Ссылки по теме

- Core utilities#lsblk

[broken link: invalid section]

- Разрешения и атрибуты файлов
- Fsck (Русский)
- fstab (Русский)
- Список приложений/Утилиты#Монтирование
- Оптический привод
- Разметка дисков
- NFS (Русский)
- NTFS-3G (Русский)
- FAT (Русский)
- QEMU (Русский)#Монтирование раздела внутри образа диска raw

[broken link: invalid section]

- Samba (Русский)
- tmpfs (Русский)
- udev (Русский)
- Udisks (Русский)
- Umask (Русский)
- USB-накопители

Состояние перевода: На этой странице представлен перевод статьи File systems. Дата последней синхронизации: 10 августа 2017. Вы можете помочь синхронизировать перевод, если в английской версии произошли изменения.

Эта страница нуждается в сопроводителе

Статья не гарантирует актуальность информации. Помогите русскоязычному сообществу поддержкой подобных страниц. См. Команда переводчиков ArchWiki

Из **Википедии**:

> Файловая система (англ. file system) — порядок, определяющий способ организации, хранения и именования данных на носителях информации в компьютерах, а также в другом электронном оборудовании: цифровых фотоаппаратах, мобильных телефонах и т.п. Файловая система определяет формат содержимого и способ физического хранения информации, которую принято группировать в виде файлов. Конкретная файловая система определяет размер имен файлов и (каталогов), максимальный возможный размер файла и раздела, набор атрибутов файла. Некоторые файловые системы предоставляют сервисные возможности, например, разграничение доступа или шифрование файлов.

Отдельные разделы дисков можно настроить с использованием одной из множества доступных файловых систем. У каждой есть свои преимущества, недостатки и уникальные особенности. Ниже приведен краткий обзор поддерживаемых файловых систем; также и ссылки на страницы Википедии, которые предоставляют гораздо больше информации.

Contents

[hide]

Типы файловых систем

Смотрите **`filesystems(5)`** для общего обзора и **Википедию:Сравнение файловых систем** для подробного сравнения функций. Файловые системы, поддерживаемые ядром, перечислены в `/proc/filesystems`.

Файловая система	Команда создания	Утилиты пользовательского пространства	Archiso[1]	Документация ядра [2]	Заметки

Файловая система	Команда создания	Утилиты пользовательского пространства	Archiso[1]	Документация ядра [2]	Заметки
Btrfs	mkfs.btrfs(8)	btrfs-progs	Да	btrfs.txt	Статус стабильности
VFAT	mkfs.vfat(8)	dosfstools	Да	vfat.txt	
exFAT	mkfs.exfat(8)	exfat-utils	Опционально	N/A (на основе FUSE)	
F2FS	mkfs.f2fs(8)	f2fs-tools	Да	f2fs.txt	Флэш-устройства
ext3	mke2fs(8)	e2fsprogs	Да (base)	ext3.txt	
ext4	mke2fs(8)	e2fsprogs	Да (base)	ext4.txt	
HFS	mkfs.hfsplus(8) [dead link 2017-11-25]	hfsprogs AUR	Опционально	hfs.txt	Файловая система MacOS
JFS	mkfs.jfs(8)	jfsutils	Да (base)	jfs.txt	
NILFS2	mkfs.nilfs2(8)	nilfs-utils	Да	nilfs2.txt	

Файловая система	Команда создания	Утилиты пользовательского пространства	Archiso[1]	Документация ядра [2]	Заметки
NTFS	mkfs.ntfs(8)	ntfs-3g	Да	N/A (на основе FUSE)	Файловая система Windows
Reiser4	mkfs.reiser4(8)	reiser4progsAUR	Нет		
ReiserFS	mkfs.reiserfs(8)	reiserfsprogs	Да (base)		
XFS	mkfs.xfs(8)	xfsprogs	Да (base)	xfs.txt xfs-delayed-logging-design.txt xfs-self-describing-metadata.txt	
ZFS		zfs-linuxAUR	Нет	N/A (порт OpenZFS)	

Примечание: У ядра есть свой собственный драйвер NTFS (смотрите ntfs.txt), но он имеет ограниченную поддержку на запись файлов.

Журналирование

Все вышеупомянутые файловые системы, за исключением ext2, FAT16/32, Btrfs и ZFS, используют ведение журнала. Журналирование обеспечивает отказоустойчивость путем регистрации изменений до того, как они будут привязаны к файловой системе. В случае сбоя системы или сбоя питания такие файловые системы быстрее возвращаются в сеть и реже становятся поврежденными. Ведение журнала происходит в выделенной области файловой системы.

Не все методы ведения журнала одинаковы. Ext3 и ext4 предлагают журналирование в режиме данных, в котором регистрируются как данные, так и метаданные, а также возможность вести журнал только изменений метаданных. Журналирование в режиме данных имеет ограничение скорости и не включено по умолчанию. В том же ключе Reiser4 предлагает так называемые "модели транзакций", которые включают в себя чистое ведение журнала (эквивалентное журнальному ведению журнала данных ext4), чистый подход копирования при записи (эквивалент по умолчанию btrfs) и комбинированный подход, который эвристически чередуется между двумя бывшими.

Примечание: Reiser4 не обеспечивает эквивалент поведения журналирования по умолчанию ext4 (только для метаданных).

Другие файловые системы обеспечивают упорядоченное ведение журнала, которое регистрирует только метаданные. Хотя все журналирование вернет файловую систему в допустимое состояние после сбоя, журналирование в режиме данных обеспечивает максимальную защиту от повреждений и потери данных. Однако есть компромисс в производительности системы, поскольку журналирование в режиме данных выполняет две операции записи: сначала в журнал, а затем на диск. При выборе типа файловой системы следует учитывать компромисс между скоростью системы и безопасностью данных.

Файловые системы, основанные на механизме копирования при записи, такие как Btrfs и ZFS, не должны использовать традиционный журнал для защиты метаданных, потому что они никогда не обновляются на месте. Хотя Btrfs все еще имеет журнальное дерево, подобное журналу, оно используется только для ускорения работы fdatasync/fsync.

Файловые системы на основе FUSE

Файловая система в пользовательском пространстве (FUSE) - это механизм для Unix-подобных операционных систем, который позволяет не-привилегированным пользователям создавать свои собственные файловые системы без редактирования кода ядра. Это достигается путем запуска кода файловой системы в *пространстве пользователя*, в то время как модуль ядра FUSE предоставляет только "мост" для реальных интерфейсов ядра.

Некоторые файловые системы на основе FUSE:

- **adbfs-git** — монтирует устройства Android, подключенные через USB.
  http://collectskin.com/adbfs/ || **adbfs-git**[AUR]

- **EncFS** — это пользовательская наращиваемая криптографическая файловая система.
  https://vgough.github.io/encfs/ || **encfs**

- **fuseiso** — монтирует ISO в качестве обычного пользователя.
  http://sourceforge.net/projects/fuseiso/ || **fuseiso**

- **gitfs** — файловая система FUSE, которая полностью интегрируется с git.
  https://www.presslabs.com/gitfs/ || **gitfs**[AUR]

- **xbfuse-git** — монтирует Xbox (360) ISO.
  http://multimedia.cx/xbfuse/ || **xbfuse-git**[AUR]

- **xmlfs** — представляет файл XML в качестве структуры каталогов для легкого доступа.
  https://github.com/halhen/xmlfs || **xmlfs**[AUR]

- **vdfuse** — монтирует образы дисков VirtualBox (VDI/VMDK/VHD).

https://github.com/muflone/virtualbox-includes || vdfuse[AUR]

Для получения дополнительной информации смотрите Википедия:Файловая система в пространстве пользователей#Примеры использования.

Штабелируемые файловые системы

- **aufs** — усовершенствованная многоуровневая файловая система унификации, объединенная файловая система на основе FUSE, полностью переписанная Unionfs, отклоненная от основной линии Linux, и вместо этого OverlayFS был объединен в ядро Linux.
http://aufs.sourceforge.net || aufs[AUR]

- eCryptfs — корпоративная криптографическая файловая система представляет собой пакет программного обеспечения для шифрования диска Linux. Он реализует шифрование на уровне файловой системы, совместимый с POSIX, с целью предложить функциональность, аналогичную функции GnuPG на уровне операционной системы.
http://ecryptfs.org || ecryptfs-utils

- **mergerfs** — объединенная файловая система на основе FUSE.
https://github.com/trapexit/mergerfs || mergerfs[AUR]

- **mhddfs** — файловая система Multi-HDD FUSE, объединенная на основе FUSE.
http://mhddfs.uvw.ru || mhddfs[AUR]

- overlayfs — это служба файловой системы для Linux, которая реализует объединение для монтирования других файловых систем.
https://www.kernel.org/doc/Documentation/filesystems/overlayfs.txt || linux

- **Unionfs** — это служба файловой системы для Linux, FreeBSD и NetBSD, которая реализует объединение монтирования для других файловых систем.
http://unionfs.filesystems.org/ || not packaged? search in AUR

- **unionfs-fuse** — реализация пользовательского пространства Unionfs.
https://github.com/rpodgorny/unionfs-fuse || unionfs-fuse

Файловые системы только для чтения

- SquashFS — сжимающая файловая система для GNU/Linux, предоставляющая доступ к данным в режиме "только для чтения". Squashfs сжимает файлы, индексные дескрипторы и каталоги, а также поддерживает блоки размером до 1024 Кбайт для лучшего сжатия.
http://squashfs.sourceforge.net/ || squashfs-tools

Кластерные файловые системы

- **Ceph** — унифицированная распределенная система хранения, предназначенная для отличной производительности, надежности и масштабируемости.
  https://ceph.com/ || `ceph`

- **Glusterfs** — кластерная файловая система способна масштабироваться до нескольких пета-байт.
  https://www.gluster.org/ || `glusterfs`

- **IPFS** — одноранговый протокол гипермедиа, чтобы сделать Интернет более быстрым, безопасным и открытым. IPFS нацелена на замену HTTP и создание лучшей сети для всех нас. Использует блоки для хранения частей файла, каждый сетевой узел хранит только интересующий контент, обеспечивает дедупликацию, распространение, масштабируемую систему, ограниченную только пользователями. (В настоящее время в aplha)
  https://ipfs.io/ || `go-ipfs`

- **MooseFS** — это отказоустойчивая, высокодоступная и высокопроизводительная сетевая распределенная файловая система.
  https://www.gluster.org/ || `moosefs`

- **OpenAFS** — реализация с открытым исходным кодом распределенной файловой системы AFS
  http://www.openafs.org || `openafs`[AUR]

- **OrangeFS** — это масштабируемая сетевая файловая система, предназначенная для прозрачного доступа к дисковой памяти на нескольких серверах параллельно. Имеет оптимизированную поддержку MPI-IO для параллельных и распределенных приложений. Упрощает использование параллельного хранения не только для клиентов Linux, но и для Windows, Hadoop и WebDAV. POSIX-совместимая. Часть ядра Linux, начиная с версии 4.6.
  http://www.orangefs.org/ || not packaged? **search in AUR**

- **Sheepdog** — распределенная система хранения объектов для объемных и контейнерных сервисов и разумно управляет дисками и узлами.
  https://sheepdog.github.io/sheepdog/ || not packaged? **search in AUR**

- **Tahoe-LAFS** — файловая система Thahoe Least-Authority - это бесплатное и открытое, безопасное, децентрализованное, отказоустойчивое, одноранговое распределенное хранилище данных и распределенная файловая система.
  https://tahoe-lafs.org/ || `tahoe-lafs`[AUR]

Определение существующих файловых систем

Чтобы определить существующие файловые системы, вы можете использовать **lsblk**:

```
$ lsblk -f

NAME   FSTYPE LABEL     UUID                                 MOUNTPOINT
sdb
└─sdb1 vfat   Transcend 4A3C-A9E9
```

Существующая файловая система, если она есть, будет показана в столбце `FSTYPE`. Если она с**монтирова**на, тогда появится в столбце `MOUNTPOINT`.

Создание файловой системы

Файловые системы обычно создаются на **раздел**е, внутри логических контейнеров, таких как **LVM**, **RAID** и **dm-crypt**, или в обычном файле (смотрите **w:Loop device**). В этом разделе описывается случай раздела.

Примечание: Файловые системы могут быть записаны непосредственно на диск, называемый **superfloppy** или *безраздельным диском*. С этим методом связаны определенные ограничения, особенно при **загрузке** с такого диска. Для примеров смотрите **Btrfs#Безраздельный диск Btrfs**[broken link: invalid section].

Важно:

- После создания новой файловой системы данные, ранее сохраненные на этом разделе, вряд ли можно будет восстановить. **Создайте резервную копию любых данных, которые вы хотите сохранить**.
- Цель данного раздела может ограничить выбор файловой системы. Например, **системный раздел EFI** должен содержать файловую систему FAT32 (`mkfs.vfat`), а файловая система, содержащая каталог `/boot`, должна поддерживаться с помощью **загрузчик**а.

Прежде чем продолжить, **определите устройство**, в котором будет создана файловая система, и независимо от того, монтируется ли она. Например:

```
$ lsblk -f

NAME   FSTYPE   LABEL       UUID                                 MOUNTPOINT
sda
├─sda1                      C4DA-2C4D
```

```
├─sda2 ext4                 5b1564b2-2e2c-452c-bcfa-d1f572ae99f2 /mnt
└─sda3                      56adc99b-a61e-46af-aab7-a6d07e504652
```

Перед продолжением **необходимо** **размонтировать** файловые системы. В приведенном выше примере существующая файловая система находится на `/dev/sda2` и монтируется в `/mnt`. Он будет размонтирован командой:

```
# umount /dev/sda2
```

Чтобы найти только смонтированные файловые системы, смотрите **#Список смонтированных файловых систем**.

Чтобы создать новую файловую систему, используйте **`mkfs(8)`**. Смотрите **#Типы файловых систем** для точного типа, а также утилиты пользовательского пространства, которые вы, возможно, захотите установить для конкретной файловой системы.

Например, чтобы создать новую файловую систему типа **ext4** (обычно для разделов данных Linux) на `/dev/sda1`, запустите:

```
# mkfs.ext4 /dev/sda1
```

Совет:

- Используйте флаг `-L` *mkfs.ext4*, чтобы указать **метку файловой системы**[broken link: invalid section]. *e2label* можно использовать для изменения метки в существующей файловой системе.
- Файловые системы могут быть *изменены* после создания с определенными ограничениями. Например, размер файловой системы **XFS** может быть увеличен, но он не может быть уменьшен. Для получения дополнительной информации смотрите **Возможности изменения размера** и соответствующую документацию файловой системы.

Новая файловая система теперь может быть смонтирована в выбранный каталог.

Монтирование файловой системы

Чтобы вручную смонтировать файловую систему, расположенную на устройстве (например, раздел) к каталогу, используйте **`mount(8)`**. В этом примере монтируется `/dev/sda1` в `/mnt`.

```
# mount /dev/sda1 /mnt
```

Это прикрепляет файловую систему раздела `/dev/sda1` в каталог `/mnt`, делая содержимое файловой системы видимым. Любые данные, существовавшие в `/mnt` перед этим действием, становятся невидимыми до тех пор, пока устройство не будет размонтировано.

fstab содержит информацию о том, как устройства должны автоматически монтироваться, если они присутствуют. Для получения дополнительной информации о том, как изменить это поведение, смотрите статью **fstab**.

Если устройство указано в `/etc/fstab`, и в командной строке указывается только устройство или точки монтирования, эта информация будет использоваться при монтирование. Например, если `/etc/fstab` содержит строку, указывающую, что `/dev/sda1` должен быть смонтирован в `/mnt`, тогда он автоматически будет монтировать это устройство к этому месту:

```
# mount /dev/sda1
```

Или

```
# mount /mnt
```

mount содержит несколько параметров, многие из которых зависят от указанной файловой системы. Параметры могут быть изменены:

- использование флагов в командной строке с *mount*
- редактирование **fstab**
- создание правил **udev**
- **самостоятельно компилировать ядро**
- или используя скрипты монтирования файловой системы (расположенные по адресу `/usr/bin/mount.*`).

Более подробную информацию смотрите в связанных статьях и статье интересующей файловой системы.

Список смонтированных файловых систем

Чтобы просмотреть все смонтированные файловые системы, используйте **`findmnt(8)`**:

```
$ findmnt
```

findmnt принимает множество аргументов, которые могут фильтровать вывод и отображать дополнительную информацию. Например, в качестве аргумента может принимать устройство или точку монтирования для отображения только информации о том, что указывается:

```
$ findmnt /dev/sda1
```

findmnt собирает информацию из `/etc/fstab`, `/etc/mtab` и `/proc/self/mounts`.

Размонтирование файловой системы

Чтобы размонтировать файловую систему, используйте **`umount(8)`**. Можно указать либо устройство, содержащее файловую систему (например, `/dev/sda1`), либо точку монтирования (например, `/mnt`):

```
# umount /dev/sda1
```

Или

```
# umount /mnt
```

Смотрите также

- **`filesystems(5)`**
- **Документация файловых систем, поддерживаемых linux**
- **Википедия:Файловая система**
- **Википедия:mount**

Categories:

- Русский
- File systems (Русский)
- Lists (Русский)