Файловые системы и типы хранения: HDD, SSD, NVMe, eMMC

1. Введение: почему тип носителя имеет значение

Выбор файловой системы в Linux нередко воспринимается как "техническая формальность" — форматировать раздел в ext4, смонтировать и забыть. Однако на практике **тип физического хранилища** — HDD, SSD, NVMe или eMMC — **прямо влияет на то, как будет вести себя файловая система**, её производительность, надёжность и срок службы устройства.

Современные носители отличаются не только скоростью, но и архитектурой доступа, ресурсами перезаписи, поддержкой дополнительных команд (например, TRIM) и глубиной очередей. Поэтому файловая система, идеально подходящая для сервера на HDD, может буквально "убить" контроллер встраиваемого устройства или тормозить на NVMe-диске.

Почему важно учитывать тип носителя

- 💽 **HDD:** хороши при последовательном доступе, но плохо справляются с мелкими и случайными записями. Выравнивание, размер блоков и стратегия журналирования здесь критичны.
- ⚡ **SSD и NVMe:** обладают высоким числом IOPS, но чувствительны к износу. Использование `discard`, выравнивания и CoW-файловых систем может как помочь, так и навредить.
- 🎁 **eMMC и SD:** обычно имеют слабый контроллер и плохо справляются с журналированием и случайной записью. Использование ext4 "по умолчанию" может ускорить деградацию памяти.

Примеры распространённых ошибок

- ✕ Включение TRIM (`discard`) на HDD — бесполезно и может замедлить операции.
- ✕ Использование Btrfs на дешёвом eMMC — ведёт к раннему износу и фризам

из-за CoW.
- ✕ Форматирование eMMC в ext4 с журналом — избыточное количество перезаписей блоков.
- ✕ Отсутствие выравнивания stripe unit при создании XFS на NVMe/RAID — снижение производительности на 30–50%.

Эта статья поможет разобраться, **как правильно подбирать файловую систему и параметры монтирования под конкретный тип хранилища**, чтобы добиться максимальной производительности, долговечности и надёжности.

2. Обзор типов хранилищ

2.1. HDD (жёсткие диски)

HDD (Hard Disk Drive) — это классический тип хранения, основанный на вращающихся магнитных пластинах и механических головках. Несмотря на возрастающее распространение SSD и NVMe, HDD остаются востребованными благодаря своей доступности и высокой ёмкости.

Особенности:

- 📄 **Последовательный доступ:** при линейном чтении и записи HDD показывают стабильную скорость.
- 🎁 **Большой объём:** до 16–20 ТБ в одном устройстве, недорого по цене за гигабайт.
- 🐌 **Медленный seek:** перемещение головки между секторами даёт значительные задержки при случайном доступе.

Жёсткие диски отлично подходят для хранения данных, которые не требуют высокой частоты доступа и не меняются часто:

- 🗄️ Архивы и резервные копии
- 📜 Лог-файлы, которые записываются последовательно
- 📄 Файловые хранилища без частого перезаписывания
- 🎞️ Медиафайлы: фото, видео, образы

Однако при неправильной настройке файловой системы производительность HDD может серьёзно пострадать. Например, использование CoW-механизмов (как в Btrfs)

или активное фрагментирование структуры каталогов ухудшает работу с данными.

💡 Рекомендуемые ФС для HDD

- **ext4** — надёжна, быстра, легко восстанавливается
- **XFS** — подходит для больших файлов и логов
- **Btrfs** — с осторожностью, если нужны снапшоты

Избегайте `discard`, TRIM и CoW, если нет особой необходимости.

2.2. SSD (SATA)

SSD (Solid State Drive) на базе SATA — это тип хранилища, использующий флеш-память (NAND) и контроллер с буфером для хранения данных без движущихся частей. Такие накопители обеспечивают **многократный выигрыш в скорости** по сравнению с HDD, особенно при случайном доступе.

Важно: SSD имеют ограниченное количество циклов записи. Это делает актуальными вопросы оптимизации файловой системы для **снижения количества перезаписей**.

Ключевые особенности SATA SSD:

- ⚡ **Быстрый доступ:** особенно к мелким файлам и при параллельных запросах
- 📉 **Ограниченный ресурс записи:** важна минимизация лишних операций
- **Необходимость TRIM:** для информирования контроллера о свободных блоках
- **Чувствительность к выравниванию:** несоответствие размеру erase-block снижает производительность

SSD отлично подходят для большинства задач, особенно на рабочих станциях, ноутбуках и серверах с высокой I/O-нагрузкой. Но без правильной настройки файловой системы можно ускорить их износ и снизить эффективность.

Рекомендации:

- Использовать опции монтирования `discard` или выполнять регулярный `fstrim`
- Отключить `atime`, использовать `noatime` или `relatime`
- Выравнивать разделы по erase block size (обычно 1 MiB)

✅ Рекомендуемые ФС для SSD (SATA)

- **ext4** — стабильность и гибкость
- **Btrfs** — снапшоты, CoW, сжатие

- **XFS** — высокая скорость при больших файлах

Используйте `fstrim` по расписанию, если не применяется `discard`.

2.3. NVMe SSD

NVMe SSD — это твердотельные накопители нового поколения, подключающиеся по интерфейсу **PCI Express** и использующие протокол `NVMe` (Non-Volatile Memory Express). Они обеспечивают **на порядок более высокую производительность** по сравнению с SATA SSD как по скорости чтения/записи, так и по числу IOPS (операций ввода-вывода в секунду).

Ключевое преимущество: минимальные задержки, многопоточность и высокая параллельность за счёт нескольких очередей команд.

Основные особенности NVMe SSD:

- 📡 **Поддержка десятков тысяч IOPS:** идеальны для баз данных и виртуализации
- **Поддержка DAX:** прямой доступ в память без использования page cache (для XFS, ext4 с опциями)
- 🔁 **Многопоточность:** несколько очередей команд вместо одной (как в AHCI/SATA)
- ⚠️ **Чувствительность к выравниванию:** плохое выравнивание снижает производительность

NVMe накопители особенно эффективны на серверах, рабочих станциях и CI/CD-инфраструктуре, где требуется быстрый отклик и высокая пропускная способность. Однако такие устройства требуют внимательной настройки файловой системы, особенно при использовании CoW и журналирования.

Рекомендации по настройке:

- Обязательно учитывать **выравнивание по stripe unit** при создании файловой системы (например, `mkfs.xfs -d su=...`)
- Использовать опции `noatime`, `logbufs=8` (XFS)
- Для прямого доступа — активировать `dax` на уровне монтирования (если поддерживается)

⚙️ Рекомендуемые ФС для NVMe

- **XFS** — высокая параллельность, DAX, масштабируемость

- **ext4** — стабильность + поддержка DAX
- **Btrfs** — при необходимости снапшотов и сжатия

Выбирайте файловую систему в зависимости от сценария — Btrfs на десктопе, XFS на сервере, ext4 — универсально.

2.4. eMMC / UFS / SD-карты

eMMC, **UFS** и **SD-карты** — это типы встроенной или сменной флеш-памяти, часто используемые в смартфонах, роутерах, встраиваемых системах, бюджетных ноутбуках и устройствах IoT. Несмотря на невысокую цену и компактность, такие носители обладают **серьёзными ограничениями**, особенно при длительной работе в роли основного хранилища.

⚠️ **Внимание:** большинство eMMC и SD-устройств используют **примитивный контроллер**, часто без DRAM-буфера и с ограниченным ресурсом перезаписи. Это делает их крайне уязвимыми к износу при неправильной файловой системе.

Типичные проблемы таких носителей:

- **Низкое качество контроллера:** неточности в реализации wear-leveling и TRIM
- 🔥 **Износ блоков:** при активной записи выходят из строя за месяцы
- 🐌 **Слабая производительность:** особенно при случайной записи
- ⚠️ **Журналируемые ФС (ext4, XFS) без настроек:** могут "убить" устройство раньше срока

Для таких типов носителей необходимо использовать **минимально избыточные файловые системы**, настроенные на редкие и последовательные записи.

Рекомендации:

- Отключить журналирование (`tune2fs -O ^has_journal` для ext4)
- Использовать опции `noatime,nodiratime,commit=600`
- Настроить периодическую TRIM через `fstrim` (если поддерживается)
- Рассмотреть не-журналируемые ФС: `F2FS`, `ext2` или ext4 без журнала

🔍 Подходящие ФС для eMMC / SD / UFS

- **F2FS** — специально для NAND-памяти
- **ext4 (без журнала)** — компромисс между совместимостью и надёжностью
- **ext2** — в системах, где стабильность важнее восстановления

Избегайте использования XFS, Btrfs и других CoW/журналируемых ФС без донастройки на таких устройствах.

3. Подбор файловой системы под носитель

Выбор подходящей файловой системы зависит не только от сценария использования, но и от типа физического хранилища. Правильная комбинация может значительно повысить надёжность, скорость и срок службы оборудования.

Тип носителя	Рекомендуемые ФС	Особенности
HDD	`ext4`, `XFS`	Простота, устойчивость, быстрая проверка (`fsck`)
SSD SATA	`ext4`, `Btrfs`	Поддержка `TRIM`, опции `noatime`, `discard`, управление износом
NVMe	`XFS`, `Btrfs`	Многопоточность, `DAX`, CoW, критично точное выравнивание
eMMC / SD	`ext2`, `F2FS`	Без журнала, адаптация под Flash, минимизация записей

💡 **Совет:** при работе с нестандартными накопителями (например, USB-флешками или китайскими eMMC) — всегда отключайте журнал и избегайте фоновых индексаций. Используйте `ext2` или `F2FS`.

4. Оптимизация параметров монтирования

Параметры монтирования позволяют тонко настроить поведение файловой системы под конкретный тип носителя. Они влияют на производительность, износ, целостность данных и отклик системы.

Общие полезные параметры

- `noatime` — не обновлять время доступа к файлу (снижает число записей)
- `relatime` — компромисс: обновляет atime только при изменении mtime
- `discard` — включает TRIM (автоочистка блоков на SSD/eMMC)
- `commit=N` — период записи журнала (в секундах)
- `barrier=0/1` — контроль барьеров записи (1 — безопасно, 0 — рискованно)

- `logbufs=8` — XFS: увеличивает число буферов для журнала

Рекомендации по типам устройств

- **SSD:** `discard,noatime,lazy_itable_init=1,commit=10`
- **NVMe:** `inode64,logbufs=8,dax,reflink=1`
- **HDD:** `commit=60,barrier=1,journal_async_commit`
- **eMMC / SD:** отключить журнал (`tune2fs -O ^has_journal`), использовать `noatime`, `commit=120`

⚠️ **Важно:** не все параметры подходят по умолчанию. Убедитесь, что файловая система и ядро поддерживают выбранные опции. Например, `dax` работает только с NVMe и XFS/Btrfs при специфических условиях.

5. Поддержка TRIM и `fstrim`

TRIM — команда, уведомляющая SSD или eMMC о том, какие блоки данных больше не используются, чтобы контроллер мог их заранее очистить. Это повышает производительность и уменьшает износ памяти.

Способы работы с TRIM

- **Параметр монтирования `discard`:**
  TRIM выполняется при каждом удалении файла. Просто, но может замедлять операции удаления.
- **Утилита `fstrim`:**
  Ручной или периодический запуск TRIM-команды на весь раздел. Более эффективно и контролируемо.

TRIM в systemd-системах

Во многих современных дистрибутивах (Ubuntu, Fedora, НАЙС.ОС) по умолчанию используется `fstrim.timer`, который запускает TRIM еженедельно:

```
systemctl status fstrim.timer
```

Чтобы запустить TRIM вручную:

```
sudo fstrim -v /
```

Можно также добавить в crontab или использовать systemd-сервисы для своих разделов.

⚠️ **Подводные камни:**

- На дешёвых SSD/eMMC TRIM может вызвать временные подвисания (особенно с `discard`).
- Некоторые контроллеры плохо реализуют TRIM, возможны проблемы с производительностью.
- На HDD TRIM не нужен — отключайте `discard`!

6. Совместимость с CoW и `reflink`

Современные файловые системы, такие как **Btrfs** и **XFS** (с опцией `reflink=1`), используют технологию **Copy-on-Write (CoW)**. Это означает, что при изменении данных создаётся новая копия блока, а старая сохраняется до подтверждения.

Преимущества CoW и reflink

- 🔄 **Атомарность:** изменения применяются полностью или не применяются вообще.
- 💾 **Снапшоты без копирования:** reflink позволяет быстро дублировать файлы, не занимая дополнительное место.
- ⚡ **На NVMe:** CoW даёт максимум преимуществ благодаря высокой скорости и параллельности.

Когда CoW вреден

- 🐢 **На eMMC/SD:** CoW может быстро изнашивать память, особенно при частых перезаписях одних и тех же файлов.
- **Слабые контроллеры:** не справляются с частыми аллокациями и фрагментацией, что снижает производительность.
- ⚠️ **При использовании SQLite, VM-дисков, Docker:** CoW мешает прямой перезаписи и снижает скорость.

В таких случаях CoW можно отключить:

```
chattr +C /path/to/directory
```

⚠️ Только на XFS и Btrfs (при создании файловой системы с `nodatacow` или через `chattr`).
💡 **Совет:** включайте `reflink=1` на **XFS** при создании, если планируется работа с снапшотами и клонированием образов:

```
mkfs.xfs -m reflink=1 /dev/sdX
```

7. Особые случаи

В некоторых сценариях выбор файловой системы и параметров монтирования должен учитывать особенности устройства или задач. Ниже — практические рекомендации для таких случаев.

USB-флешки и внешние накопители

- **ext4:** лучший вариант, если устройство используется только в Linux. Рекомендуется отключить журнал: `-O ^has_journal`.
- **exFAT:** кросс-платформенный, поддерживается в Windows, Android и Linux (через `exfatprogs`).
- **NTFS:** совместимость с Windows, но медленный и ресурсоёмкий. Лучше избегать на слабых устройствах.

🔒 Read-only root на eMMC

На слабых встраиваемых системах (eMMC/SD) рекомендуется монтировать корень в режиме `read-only` для защиты от износа и повреждений:

```
mount -o ro /dev/mmcblk0p2 /
```

- /var и /tmp монтируются отдельно в tmpfs или ext4 (rw).
- Обновления производятся через переключение rootfs (A/B boot или overlayfs).

Использование `ZRAM` и `tmpfs` на слабом хранилище

Если устройство имеет мало RAM и медленное хранилище, можно использовать сжатую оперативную память:

- **ZRAM:** создаёт сжатый swap в RAM. Повышает отзывчивость без износа диска.
- **tmpfs:** временные каталоги (/tmp, /var/tmp) можно монтировать в память:

```
tmpfs  /tmp  tmpfs  defaults,noatime,size=256M  0  0
```

💡 Использование `readonly root + tmpfs + zram` — популярное решение в защищённых и встраиваемых системах с eMMC.

8. Выводы и рекомендации

Файловая система должна подбираться с учётом типа накопителя.
Универсального решения не существует — и слепой выбор ext4 "по привычке" может привести к потере производительности или ресурса устройства.

- ✅ **Не выбирайте ext4 «по умолчанию»** — оцените устройство, его роль и срок службы.
- 🔍 **Не все SSD одинаковы:** NVMe требует других подходов (выравнивание, DAX), чем SATA или eMMC.
- **Во встраиваемых системах** надёжность и долговечность важнее скорости — используйте `F2FS`, `ext2`, отключение журнала и `readonly root`.
- 🛠️ **Настраивайте параметры монтирования**: `noatime`, `commit`, `discard` и другие критически влияют на поведение ФС.

📚 Таблица совместимости файловых систем и типов хранилищ

Хранилище	ext4	XFS	Btrfs	F2FS	ReiserFS	exFAT
HDD	✅	✅	✅	⚠️	⚠️	⛔
SSD SATA	✅	✅	✅	⚠️	⛔	⛔
NVMe	⚠️	✅	✅	⛔	⛔	⛔

Хранилище	ext4	XFS	Btrfs	F2FS	ReiserFS	exFAT
eMMC	⚠️	⛔	⛔	✅	⛔	⚠️
USB flash	✅	⚠️	⚠️	⚠️	⛔	✅

✅ — рекомендовано, ⚠️ — допустимо с ограничениями, ⛔ — не рекомендуется.