Как проверить и восстановить файловую систему

1. Введение

Файловая система — это структура, которая обеспечивает доступ, запись и хранение данных на диске. При сбоях питания, аппаратных ошибках, программных сбоях или некорректном отключении питания она может выйти из строя, что приведёт к потере данных или невозможности загрузки системы.

❗ Важно: при любых подозрениях на повреждение ФС **нельзя продолжать работу как обычно** — это может усугубить проблему.

🧮 Зачем проверять файловую систему?

- 💥 После **резкого отключения питания** или зависания системы;
- 🐞 При возникновении **I/O ошибок** в журналах (`dmesg`, `journalctl`);
- При **монтировании раздела в режиме read-only** без запроса пользователя;
- 🐌 При **заметном падении производительности** доступа к диску.

🔐 Когда проверка обязательна

Если при загрузке вы видите сообщение о невозможности монтировать раздел, либо система предлагает запустить `fsck` — откладывать нельзя.

⚠️ **Общее правило:** не запускайте проверку на смонтированной файловой системе — это может привести к разрушению метаданных.
Исключение: `Btrfs` поддерживает онлайн-проверку с помощью `btrfs scrub`.

В этой статье мы рассмотрим пошагово, как безопасно проверять и восстанавливать файловые системы **ext4**, **XFS** и **Btrfs**, используя штатные утилиты Linux.

2. Проверка и восстановление ext4: `fsck`, `e2fsck`

Для файловых систем **ext2/ext3/ext4** используется классическая утилита `fsck`, с

фактической реализацией через `fsck.ext4` или `e2fsck`. Перед запуском утилиты крайне важно размонтировать раздел, чтобы избежать повреждения данных.

📥 Размонтирование перед проверкой

Обязательно размонтируйте файловую систему перед проверкой:

```
sudo umount /dev/sdX1
```

Если раздел используется системой, рекомендуется загрузиться с LiveCD.

Проверка файловой системы

Запуск проверки с исправлением ошибок:

```
sudo fsck.ext4 /dev/sdX1
```

Или эквивалентная команда:

```
sudo e2fsck /dev/sdX1
```

⚙️ Полезные флаги

- `-n` — только проверка (read-only), без изменений:

```
sudo fsck.ext4 -n /dev/sdX1
```

`-y` — автоматически подтверждать исправление всех ошибок:

```
sudo fsck.ext4 -y /dev/sdX1
```

💡 **Примечание:** `fsck.ext4` и `e2fsck` — это одно и то же. Первый — удобный обёртка, второй — низкоуровневая утилита.

🔧 Особенности ext4

- Благодаря журналированию **проверка ext4 обычно быстрая**, если не было жёстких сбоев.
- Иногда достаточно лишь выполнить `fsck -n`, чтобы убедиться в целостности.

🔁 Автоматическая проверка при загрузке

Параметры периодической проверки можно настроить с помощью `tune2fs`:

```
# Проверять каждые 30 монтирований или 1 месяц
sudo tune2fs -c 30 -i 1m /dev/sdX1
```

Также важно правильно указать приоритет проверки в `/etc/fstab`:

```
/dev/sdX1  /  ext4  defaults  0 1
```

⚠️ **Не запускайте fsck на смонтированном разделе**, особенно с правами записи — это может повредить структуру ФС.

3. Проверка и восстановление XFS: `xfs_repair`

Файловая система **XFS** использует собственный инструмент для проверки и восстановления — `xfs_repair`. Утилита `fsck` не применима к XFS и выдаст предупреждение при попытке запуска.

⚠️ **Важно:** файловая система XFS должна быть **размонтирована** перед выполнением проверки. На смонтированной XFS запуск `xfs_repair` невозможен.

📥 Подготовка: размонтирование

```
sudo umount /dev/sdX1
```

Диагностика (только проверка)

Для безопасного анализа состояния файловой системы без внесения изменений используйте флаг `-n`:

```
sudo xfs_repair -n /dev/sdX1
```

Этот режим только анализирует состояние, не внося правок.

🔧 Восстановление (внесение исправлений)

Если были обнаружены проблемы, выполните полноценную проверку с

восстановлением:

```
sudo xfs_repair /dev/sdX1
```

📌 Особенности и поведение

- ⚡ Очень быстрая проверка — подходит для больших файловых систем (терабайты и выше);
- 🔄 Проверка ориентирована на **структуру метаданных** — пользовательские данные могут быть потеряны, если они повреждены;
- ⚠ При критических сбоях может потребоваться удаление повреждённых каталогов или инодов.

💡 **Рекомендация:** перед запуском `xfs_repair` создайте резервную копию или снимок раздела (например, с помощью `dd`), если есть подозрение на аппаратные ошибки.

4. Проверка и восстановление Btrfs: `btrfs check`

Файловая система **Btrfs** устроена иначе, чем ext4 и XFS: она автоматически восстанавливается за счёт контрольных сумм и встроенных механизмов целостности. Тем не менее, в случае ошибок Btrfs предлагает собственные инструменты проверки и ремонта.

Оффлайн-проверка

Проверка раздела должна выполняться **на размонтированной** файловой системе:

```
sudo umount /dev/sdX1
sudo btrfs check /dev/sdX1
```

⚠️ **Важно:** никогда не запускайте `btrfs check` на смонтированном разделе — это может привести к повреждениям.

✔ Онлайн-проверка: `btrfs scrub`

Btrfs поддерживает безопасную проверку целостности "на живую" с помощью `scrub`:

```
sudo btrfs scrub start -Bd /mountpoint
```

- `-B` — ждать завершения процесса (блокирующий режим);
- `-d` — выводить прогресс по данным.

Проверить результаты скраба:

```
sudo btrfs scrub status /mountpoint
```

Режим восстановления (только в крайних случаях)

Если файловая система серьёзно повреждена, можно использовать флаг `--repair`:

```
sudo btrfs check --repair /dev/sdX1
```

❗ Предупреждение: параметр `--repair` может повредить данные. Используйте только как последнее средство и **только после резервного копирования**.

📊 Дополнительные полезные команды

- `btrfs device stats /` — показывает статистику ошибок на устройствах;
- `btrfs filesystem show` — выводит список Btrfs-томов и устройств;
- `btrfs subvolume list /` — проверка структуры подтомов (можно диагностировать утерю метаданных);

💡 **Btrfs** — единственная штатная ФС в Linux, которая поддерживает регулярную фоновую проверку без остановки системы благодаря `scrub`.

5. Что делать, если файловая система не монтируется

Иногда система отказывается монтировать раздел: ссылается на ошибки, предлагает запустить `fsck`, или просто «зависает» при старте. В таком случае необходимо действовать осторожно, не усугубляя проблему.

🕶 Использование LiveCD/LiveUSB

Самый безопасный способ восстановления — загрузиться с Live-среды (например, Ubuntu Live, SystemRescueCD, Fedora Live) и работать с разделами «снаружи».

```
# Подключение Live-среды и открытие терминала
sudo lsblk       # посмотреть устройства
sudo fdisk -l    # уточнить разметку
```

Анализ логов ядра

Ошибки монтирования часто сопровождаются сообщениями в логе ядра. Проверить их можно через:

```
dmesg | tail -n 50
journalctl -xb | grep -i error
```

Сообщения вида `EXT4-fs error` или `XFS (sda1): internal error` помогут определить источник сбоя.

🔍 Временное монтирование в `read-only` для диагностики

Чтобы безопасно исследовать содержимое проблемного раздела (например, скопировать данные), можно примонтировать его в режиме «только для чтения»:

```
sudo mount -o ro /dev/sdX1 /mnt
```

Если монтирование прошло успешно — проверьте целостность вручную, скопируйте важные данные и затем проведите восстановление.

Проверка состояния диска (SMART)

Иногда проблема кроется в физическом износе или сбоях устройства хранения. Используйте `smartctl` для диагностики:

```
sudo smartctl -a /dev/sdX
```

- Проверьте наличие `Reallocated_Sector_Ct`, `Pending_Sector`, `Offline_Uncorrectable`
- Обратите внимание на `Power_On_Hours` и `Wear_Leveling_Count` для SSD

⚠️ **Если SMART сообщает об ошибках**, восстановление данных должно проводиться с созданием образа диска (`ddrescue`) — не используйте `fsck` на физически повреждённом диске напрямую.

6. Автоматическая проверка при загрузке

В большинстве Linux-дистрибутивов предусмотрена автоматическая проверка

ext2/ext3/ext4 при загрузке системы. Она может быть инициирована при определённом количестве монтирований, времени с последней проверки или по флагу в `/etc/fstab`.

⚙️ Настройка через `tune2fs`

Можно задать интервалы принудительной проверки для ext4:

```
sudo tune2fs -c 30 -i 1m /dev/sdX1
```

- `-c 30` — проверять каждые 30 монтирований;
- `-i 1m` — проверять не реже одного раза в месяц (`1d`, `1w`, `6m` и т.п.).

Проверить текущие значения можно так:

```
sudo tune2fs -l /dev/sdX1 | grep -i 'mount count\|check interval'
```

💡 **Рекомендация:** для важных систем лучше настроить интервал по времени, а не по количеству монтирований (если система работает долго без перезагрузок).

📄 Параметр `fs_passno` в `/etc/fstab`

Финальный шаг — указать приоритет проверки в файле `/etc/fstab`:

```
/dev/sdX1  /     ext4  defaults  0 1
/dev/sdX2  /home  ext4  defaults  0 2
```

- Последнее число — **fs_passno**:
- `1` — корневая ФС, проверяется первой;
- `2` — второстепенные ФС (например, `/home`);
- `0` — не проверяется при загрузке.

Важно: автоматическая проверка выполняется только при использовании `initramfs` и соответствующих сервисов загрузчика (например, systemd или init).

7. Заключение

Для каждой файловой системы в Linux предусмотрены **свои уникальные инструменты** проверки и восстановления. Они **не взаимозаменяемы** и не совместимы между собой:

- `fsck` и `e2fsck` — только для **ext2/ext3/ext4**
- `xfs_repair` — только для **XFS**
- `btrfs check` и `scrub` — только для **Btrfs**

⚠️ **Никогда не запускайте восстановительные утилиты на неподходящей ФС.** Это может привести к полной потере данных.

✔ Профилактика — лучшее восстановление

- Регулярно проверяйте файловую систему в фоне или при загрузке (особенно для ext4);
- Перед запуском любых команд восстановления — **делайте резервную копию** раздела или создайте образ через `dd`;
- Используйте LiveCD для безопасной работы с "упавшими" разделами;
- Анализируйте SMART и аппаратное состояние носителя при систематических сбоях.

💡 **Совет:** если данные критичны — работайте не с оригиналом, а с копией (например, через `ddrescue`).

📚 Приложение: таблица инструментов по типу файловой системы

Файловая система	Проверка	Восстановление	Онлайн-проверка	Примечания
ext4	`fsck.ext4`, `e2fsck`	`fsck -y`	✕	Безопасна и надёжна
XFS	`xfs_repair -n`	`xfs_repair`	✕	Только офлайн-проверка
Btrfs	`btrfs check`	`btrfs check --repair`	✓ (через `scrub`)	Поддерживает онлайн-проверку

Соблюдая простые правила диагностики и восстановления, вы минимизируете риски потери данных и повышаете надёжность работы Linux-системы в целом.