Red Hat Enterprise Linux 6

GFS 2

Red Hat GFS 2

Редакция 7

Last Updated: 2017-10-13

Red Hat Enterprise Linux 6 GFS 2

Red Hat GFS 2
Редакция 7

Юридическое уведомление

Аннотация

В этом документе обсуждаются вопросы настройки и эксплуатации файловой системы GFS2 в Red Hat Enterprise Linux 6.

Содержание

ВВЕДЕНИЕ **5**
1. ЦЕЛЕВАЯ АУДИТОРИЯ 5
2. ДОПОЛНИТЕЛЬНАЯ ДОКУМЕНТАЦИЯ 5
3. ОТЗЫВЫ И ПРЕДЛОЖЕНИЯ 5

ГЛАВА 1. ОБЗОР GFS2 . . . . . **7**
1.1. НОВЫЕ И ИЗМЕНЕННЫЕ ФУНКЦИИ 8
1.1.1. Red Hat Enterprise Linux 6 8
1.1.2. Red Hat Enterprise Linux 6.1 8
1.1.3. Red Hat Enterprise Linux 6.2 8
1.1.4. Red Hat Enterprise Linux 6.3 9
1.1.5. Red Hat Enterprise Linux 6.4 9
1.1.6. Red Hat Enterprise Linux 6.6 9
1.2. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ GFS2 9
1.3. УСТАНОВКА GFS2 10
1.4. ОСНОВНЫЕ ОТЛИЧИЯ GFS2 ОТ GFS 10
1.4.1. Команды GFS2 10
1.4.2. Другие различия 11
1.4.2.1. Контекстные ссылки 11
1.4.2.2. gfs2.ko 11
1.4.2.3. Квоты в GFS2 12
1.4.2.4. Ведение журналов 12
1.4.2.5. Динамическое добавление журналов 12
1.4.2.6. atime_quantum 12
1.4.2.7. mount data=ordered/writeback 12
1.4.2.8. gfs2_tool 12
1.4.2.9. gfs2_edit 13
1.4.3. Производительность 13

ГЛАВА 2. ФУНКЦИОНАЛЬНЫЕ ОСОБЕННОСТИ И КОНФИГУРАЦИЯ GFS2 . . . . . **15**
2.1. ОСОБЕННОСТИ ФОРМАТИРОВАНИЯ 15
2.1.1. Размер: чем меньше, тем лучше 15
2.1.2. Размер блоков 15
2.1.3. Число журналов 16
2.1.4. Размер журналов 16
2.1.5. Группы ресурсов 16
2.2. ФРАГМЕНТАЦИЯ 17
2.3. ВЫДЕЛЕНИЕ БЛОКОВ 17
2.3.1. Оставьте свободное место 17
2.3.2. Выделение блоков владельцем файла 17
2.3.3. Предварительное выделение места 18
2.4. ПРОЕКТИРОВАНИЕ КЛАСТЕРА 18
2.5. ОСОБЕННОСТИ ЭКСПЛУАТАЦИИ 18
2.5.1. Noatime и nodiratime 18
2.5.2. Размер таблиц DLM 18
2.5.3. Виртуальная файловая система 18
2.5.4. SELinux 19
2.5.5. NFS и GFS2 19
2.5.6. Samba и GFS2 20
2.6. РЕЗЕРВНОЕ КОПИРОВАНИЕ 20
2.7. АППАРАТНЫЕ ОСОБЕННОСТИ 21
2.8. ПОРТАЛ ПОЛЬЗОВАТЕЛЕЙ RED HAT 21

2.9. БЛОКИРОВКА УЗЛОВ GFS2 21
2.9.1. Блокировка Posix 22
2.9.2. Оптимизация производительности 23
2.9.3. Статистика блокировок 23

ГЛАВА 3. НАЧАЛО РАБОТЫ 27
3.1. ПРЕДВАРИТЕЛЬНЫЕ ТРЕБОВАНИЯ 27
3.2. ПЕРВОНАЧАЛЬНАЯ КОНФИГУРАЦИЯ 27

ГЛАВА 4. УПРАВЛЕНИЕ GFS2 29
4.1. СОЗДАНИЕ ФАЙЛОВОЙ СИСТЕМЫ 29
4.1.1. Формат команд 29
4.1.2. Примеры 31
4.1.3. Полный список параметров 31
4.2. МОНТИРОВАНИЕ ФАЙЛОВОЙ СИСТЕМЫ 32
4.2.1. Формат команд 33
4.2.2. Пример 33
4.2.3. Полная форма 33
4.3. ОТКЛЮЧЕНИЕ ФАЙЛОВОЙ СИСТЕМЫ 36
4.3.1. Формат команд 36
4.4. ОСОБЕННОСТИ ПОДКЛЮЧЕНИЯ GFS2 36
4.5. УПРАВЛЕНИЕ КВОТАМИ В GFS2 36
4.5.1. Настройка дисковых квот 37
4.5.1.1. Настройка квот 37
4.5.1.1.1. Формат команд 37
4.5.1.1.2. Примеры 38
4.5.1.2. Создание базы данных квот 38
4.5.1.3. Установка квот для пользователей 38
4.5.1.4. Установка квот для групп пользователей 39
4.5.2. Repquota 39
4.5.3. Quotacheck 40
4.5.4. Quotasync 40
4.5.4.1. Формат команд 41
4.5.4.2. Примеры 41
4.5.5. Дополнительные ресурсы 42
4.6. УВЕЛИЧЕНИЕ РАЗМЕРА ФАЙЛОВОЙ СИСТЕМЫ 42
4.6.1. Формат команд 42
4.6.2. Комментарии 42
4.6.3. Примеры 42
4.6.4. Полная форма 43
4.7. ДОБАВЛЕНИЕ ЖУРНАЛОВ 43
4.7.1. Формат команд 44
4.7.2. Примеры 44
4.7.3. Полная форма 44
4.8. ЖУРНАЛИРОВАНИЕ ДАННЫХ 45
4.9. ОБНОВЛЕНИЕ ATIME 46
4.9.1. Relatime 46
4.9.1.1. Формат команд 46
4.9.1.2. Пример 47
4.9.2. Noatime 47
4.9.2.1. Формат команд 47
4.9.2.2. Пример 47
4.10. ВРЕМЕННАЯ ОСТАНОВКА ФАЙЛОВОЙ СИСТЕМЫ 47

4.10.1. Формат команд 47
4.10.2. Примеры 48
4.11. ПРОВЕРКА ФАЙЛОВОЙ СИСТЕМЫ 48
4.11.1. Формат команд 49
4.11.2. Пример 49
4.12. MOUNT --BIND И КОНТЕКСТНЫЕ ССЫЛКИ 50
4.13. MOUNT --BIND И ПОРЯДОК ПОДКЛЮЧЕНИЯ 51
4.14. ФУНКЦИИ ОТЗЫВА GFS2 53

ГЛАВА 5. ДИАГНОСТИКА КОНФЛИКТОВ GFS2 55
5.1. НИЗКАЯ ПРОИЗВОДИТЕЛЬНОСТЬ GFS2 55
5.2. ЗАВИСАНИЕ GFS2 С НЕОБХОДИМОСТЬЮ ПЕРЕЗАГРУЗКИ НА ОДНОМ УЗЛЕ 55
5.3. ЗАВИСАНИЕ GFS2 С НЕОБХОДИМОСТЬЮ ПЕРЕЗАГРУЗКИ НА ВСЕХ УЗЛАХ 55
5.4. НЕ УДАЕТСЯ ПОДКЛЮЧИТЬ GFS2 НА НОВОМ УЗЛЕ КЛАСТЕРА 56
5.5. ЗАНЯТОЕ ПРОСТРАНСТВО В ПУСТОЙ ФАЙЛОВОЙ СИСТЕМЕ 56

ГЛАВА 6. GFS2 В КЛАСТЕРЕ PACEMAKER 57

ПРИЛОЖЕНИЕ A. GFS2_QUOTA 59
A.1. НАСТРОЙКА КВОТ 59
A.1.1. Формат команд 59
A.1.2. Примеры 60
A.2. СТАТИСТИКА ИСПОЛЬЗОВАНИЯ ПРОСТРАНСТВА 60
A.2.1. Формат команд 60
A.2.2. Вывод gfs2_quota 60
A.2.3. Комментарии 61
A.2.4. Примеры 61
A.3. СИНХРОНИЗАЦИЯ КВОТ 61
A.3.1. Формат команд 61
A.3.2. Примеры 62
A.4. ВКЛЮЧЕНИЕ И ОТКЛЮЧЕНИЕ КВОТ 62
A.4.1. Формат команд 62
A.4.2. Примеры 63
A.5. СТАТИСТИКА ИСПОЛЬЗОВАНИЯ КВОТ 63
A.5.1. Формат команд 63
A.5.2. Пример 63

ПРИЛОЖЕНИЕ B. ПРЕОБРАЗОВАНИЕ GFS В GFS2 64
B.1. ПРЕОБРАЗОВАНИЕ КОНТЕКСТНЫХ ССЫЛОК 64
B.2. ПОРЯДОК ПРЕОБРАЗОВАНИЯ GFS В GFS2 64

ПРИЛОЖЕНИЕ C. МОНИТОРИНГ СОБЫТИЙ И ФАЙЛ GLOCKS 66
C.1. КАТЕГОРИИ СОБЫТИЙ ТРАССИРОВКИ В GFS2 66
C.2. ОБРАБОТЧИКИ 66
C.3. GLOCKS 67
C.4. ПРОСМОТР СТАТУСА GLOCK СРЕДСТВАМИ DEBUGFS 68
C.5. ЗАПРОСЫ БЛОКИРОВКИ 71
C.6. ОБРАБОТЧИКИ СОБЫТИЙ GLOCK 72
C.7. ОТСЛЕЖИВАНИЕ БЛОЧНЫХ ОПЕРАЦИЙ 73
C.8. ОБРАБОТЧИКИ СОБЫТИЙ ЖУРНАЛОВ 73
C.9. СТАТИСТИКА GLOCK 73
C.10. ДОПОЛНИТЕЛЬНЫЕ РЕСУРСЫ 74

ПРИЛОЖЕНИЕ D. ИСТОРИЯ ПЕРЕИЗДАНИЯ 75

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ **77**

ВВЕДЕНИЕ

В этом документе приведена информация о настройке и поддержке Red Hat GFS2 (Global File System 2), которая входит в комплект Resilient Storage.

1. ЦЕЛЕВАЯ АУДИТОРИЯ

Этот документ ориентирован на администраторов Linux, имеющих навыки:

- администрирования Linux, в том числе конфигурации ядра;
- установки и конфигурации сетей с общим хранилищем, таких как Fibre Channel SAN.

2. ДОПОЛНИТЕЛЬНАЯ ДОКУМЕНТАЦИЯ

Подробная информация о Red Hat Enterprise Linux может быть найдена в руководствах:

- *Руководство по установке Red Hat Enterprise Linux 6*
- *Руководство по развертыванию* предоставляет информацию по установке, настройке и администрированию Red Hat Enterprise Linux 6.
- *Руководство по управлению накопителями* расскажет об эффективном управлении устройствами хранения данных и файловыми системами в Red Hat Enterprise Linux 6.

Информацию о комплектах High Availability и Resilient Storage можно найти в следующих документах:

- *Обзор комплекта High Availability*.
- *Управление кластером* содержит информацию об установке, настройке и управлении кластерными компонентами Red Hat.
- *Администрирование LVM* содержит информацию об управлении логическими томами, включая сведения о работе LVM в кластерном окружении.
- *DM Multipath* предоставляет информацию о многопутевых возможностях Red Hat Enterprise Linux.
- *Распределение нагрузки* предоставляет информацию о настройке высокопроизводительных систем и служб в группе физических серверов с помощью Load Balancer.
- *Примечания к выпуску* предлагают краткий обзор последнего выпуска Red Hat.

Полный каталог документации в форматах HTML, PDF и RPM можно найти на диске документации Red Hat Enterprise Linux и на сайте https://access.redhat.com/site/documentation/.

3. ОТЗЫВЫ И ПРЕДЛОЖЕНИЯ

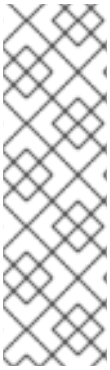
Если вы нашли опечатки или хотите оставить отзыв, создайте запрос в Bugzilla по адресу http://bugzilla.redhat.com/, выбрав компонент **doc-Global_File_System_2**. Дополнительно укажите идентификатор документа:

```
rh-gfs2(EN)-6 (2014-10-8T15:15)
```

Если у вас есть предложения по улучшению руководства, постарайтесь подробно их описать. Если же вы нашли ошибку, укажите номер раздела и окружающий текст для облегчения ее идентификации.

ГЛАВА 1. ОБЗОР GFS2

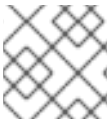
Red Hat GFS2 входит в состав комплекта Resilient Storage и напрямую взаимодействует с интерфейсом файловой системы на уровне ядра Linux (уровень VFS). В кластере GFS2 использует распределенные метаданные и журналы.



ПРИМЕЧАНИЕ

GFS2 можно развернуть в отдельной системе или как часть кластерной структуры, но Red Hat Enterprise Linux 6 поддерживает только кластерную реализацию. Red Hat поддерживает другие высокопроизводительные файловые системы, специально предназначенные для работы на индивидуальных узлах, поэтому при наличии лишь одного узла рекомендуется выбрать их, а не GFS2.

Поддержка GFS2 на автономных узлах ограничивается подключением снимков кластерных файловых систем с целью создания резервных копий.



ПРИМЕЧАНИЕ

Red Hat не поддерживает GFS2 в кластерах, где число узлов превышает 16.

В основу GFS2 положена 64-разрядная архитектура, что теоретически позволяет создать файловую систему размером 8 эксабайт. Но максимально подерживаемый размер GFS2 на сегодняшний день составляет 100 терабайт для 64-разрядной архитектуры и 16 терабайт для 32-разрядной. Если требуется больше места, обратитесь к консультанту Red Hat.

При предварительной оценке размера файловой системы обдумайте, как вы будете восстанавливать данные в случае ее сбоя. Например, проверка **`fsck.gfs2`** в большой файловой системе может занять продолжительное время, при этом используя большой объем памяти. В случае сбоя диска время восстановления также зависит от скорости резервного носителя (см. Раздел 4.11, «Проверка файловой системы»).

В распределенной схеме узлы Red Hat GFS2 можно настроить с помощью инструментов, входящих в комплект High Availability. Red Hat GFS2 обеспечивает совместный доступ к данным из кластера Red Hat; при этом представление файловой системы одинаково для всех узлов GFS2. Это позволяет процессам на разных узлах использовать файлы в GFS2 аналогично тому, как это делается в локальной системе. Подробную информацию можно найти в документе под названием *Управление кластером Red Hat*.

В то время как использование GFS2 за пределами LVM не запрещено, Red Hat официально поддерживает только файловые системы GFS2, созданные на логических томах CLVM (CLVM входит в комплект Resilient Storage). CLVM — кластерная реализация LVM, позволяющая управлять логическими томами в кластере с помощью **`clvmd`**. Подробную информацию можно найти в документе под названием *Администрирование LVM*.

За реализацию GFS2 отвечает модуль ядра **`gfs2.ko`**, который должен быть загружен на узлах кластера.

ПРИМЕЧАНИЕ

При настройке GFS2 в кластере необходимо открыть доступ к общему хранилищу. При этом не требуется монтировать GFS2 на каждом узле.

В этой главе представлены основные сведения о GFS2:

- Раздел 1.1, «Новые и измененные функции»
- Раздел 1.2, «Проектирование структуры GFS2»
- Раздел 1.4, «Основные отличия GFS2 от GFS»
- Раздел 1.3, «Установка GFS2»
- Раздел 2.9, «Блокировка узлов GFS2»

1.1. НОВЫЕ И ИЗМЕНЕННЫЕ ФУНКЦИИ

Эта секция содержит перечень новых и измененных функций GFS2 в разных выпусках Red Hat Enterprise Linux 6.

1.1.1. Red Hat Enterprise Linux 6

Далее перечислены особенности GFS2 в Red Hat Enterprise Linux 6.0.

- Использование GFS2 в индивидуальных системах не поддерживается.
- Для преобразования GFS в GFS2 используется **gfs2_convert** (см. Приложение B, *Преобразование GFS в GFS2*).
- Поддерживаются параметры монтирования: **discard**, **nodiscard**, **barrier**, **nobarrier**, **quota_quantum**, **statfs_quantum**, **statfs_percent** (см. Раздел 4.2, «Монтирование файловой системы»).
- В руководство Red Hat Enterprise Linux 6 добавлен Раздел 2.9, «Блокировка узлов GFS2».

1.1.2. Red Hat Enterprise Linux 6.1

Далее перечислены особенности GFS2 в Red Hat Enterprise Linux 6.1.

- Добавлена поддержка квотирования (см. Раздел 4.5, «Управление квотами в GFS2»).

  В предыдущих выпусках управление квотами осуществлялось с помощью **gfs2_quota** (см. Приложение A, ***gfs2_quota***).

- В текущую версию документа добавлена Глава 5, *Диагностика конфликтов GFS2*.
- Это руководство включает множество коррекций, актуальных для Red Hat Enterprise Linux 6.1.

1.1.3. Red Hat Enterprise Linux 6.2

Ниже перечислены основные изменения в Red Hat Enterprise Linux 6.2.

- **tunegfs2** выполняет некоторые функции команды **gfs2_tool**. Подробную информацию можно найти на справочной странице **tunegfs2**.

  Обновление документации:

- Раздел 4.5.4, «**Quotasync**» и Раздел A.3, «Синхронизация квот» расскажут об изменении значения **quota_quantum** (по умолчанию равно 60 секундам) при помощи параметра **quota_quantum=интервал**.
- Раздел 4.10, «Временная остановка файловой системы» включает описание команды **dmsetup suspend**.

- Приложение C, *Мониторинг событий и файл glocks* ориентировано на опытных пользователей, содержит описание интерфейса **debugfs** и обработчиков событий GFS2.

1.1.4. Red Hat Enterprise Linux 6.3

Добавлена Глава 2, *Функциональные особенности и конфигурация GFS2* с рекомендациями по оптимизации производительности GFS2.

Кроме того, внесены другие незначительные изменения и дополнения.

1.1.5. Red Hat Enterprise Linux 6.4

Глава 2, *Функциональные особенности и конфигурация GFS2* обновлена и дополнена.

1.1.6. Red Hat Enterprise Linux 6.6

Добавлена Глава 6, *GFS2 в кластере Pacemaker* с инструкциями по настройке файловой системы в кластере Pacemaker.

Кроме того, внесены другие незначительные изменения и дополнения.

1.2. ПРОЕКТИРОВАНИЕ СТРУКТУРЫ GFS2

Прежде чем приступить к установке GFS2, надо продумать ее структуру:

Узлы GFS2

Определите, какие узлы в кластере будут подключать GFS2.

Число файловых систем

Сколько файловых систем GFS2 вы создадите? Файловые системы также можно будет добавить позднее.

Имя файловой системы

Имя файловой системы должно быть уникально среди файловых систем **lock_dlm** в кластере. В примерах команд в этом документе используются имена **mydata1** и **mydata2**.

Журналы

Каждый узел должен иметь свой журнал. Если вспоследствии новые серверы начнут подключать GFS2, журналы можно будет добавить динамически (см. Раздел 4.7, «Добавление журналов»).

Диски и разделы

Идентифицируйте устройства и разделы, на основе которых будут создаваться логические тома (с помощью CLVM).

ПРИМЕЧАНИЕ

Большое число запросов создания и удаления файлов с разных узлов к одному каталогу может отрицательно сказаться на производительности. В таких случаях для каждого узла попробуйте выделить специальные каталоги для создания и удаления файлов.

Глава 2, *Функциональные особенности и конфигурация GFS2* содержит другие рекомендации.

1.3. УСТАНОВКА GFS2

Подготовьте окружение, установив комплект Red Hat High Availability. Затем можно будет установить **gfs2-utils** для GFS2 и **lvm2-cluster** для CLVM (Clustered Logical Volume Manager). Оба пакета включены в канал ResilientStorage.

Команда установки перечисленных компонентов:

```
# yum install rgmanager lvm2-cluster gfs2-utils
```

Информацию о создании кластера можно найти в документе под названием *Администрирование кластера*.

1.4. ОСНОВНЫЕ ОТЛИЧИЯ GFS2 ОТ GFS

В этой секции перечислены основные изменения, которые отличают GFS2 от GFS.

Преобразование GFS в GFS2 осуществляется с помощью **gfs2_convert** (см. Приложение B, *Преобразование GFS в GFS2*).

1.4.1. Команды GFS2

В целом, функциональность GFS2 аналогична GFS. Главное отличие состоит в том, что названия команд, которые раньше имели приставку «gfs», теперь содержат «gfs2» (см. Таблица 1.1, «Команды GFS и GFS2»).

Таблица 1.1. Команды GFS и GFS2

GFS	GFS2	Описание
mount	**mount**	Подключение файловой системы. Тип (GFS или GFS2) определяется автоматически. Список параметров команды можно найти на справочной странице gfs2_mount(8).
umount	**umount**	Отключение файловой системы.
fsck **gfs_fsck**	**fsck** **fsck.gfs2**	Проверяет и исправляет отключенную файловую систему.
gfs_grow	**gfs2_grow**	Увеличение размера файловой системы.

GFS	GFS2	Описание
gfs_jadd	**gfs2_jadd**	Добавление журнала.
gfs_mkfs **mkfs -t gfs**	**mkfs.gfs2** **mkfs -t gfs2**	Создание файловой системы.
gfs_quota	**gfs2_quota**	Управление квотами пространства в подключенной файловой системе. Начиная с Red Hat Enterprise Linux 6.1 поддерживаются стандартные функции квот Linux (см. Раздел 4.5, «Управление квотами в GFS2»).
gfs_tool	**tunegfs2** параметры mount **dmsetup suspend**	Настройка и сбор статистики. Команда **tunegfs2** была впервые представлена в Red Hat Enterprise Linux 6.2. **gfs2_tool** также поддерживается.
gfs_edit	**gfs2_edit**	Изменение внутренней структуры файловой системы. **gfs2_edit** подходит для обеих версий GFS.
gfs_tool setflag jdata/inherit_jdata	**chattr +j** (рекомендуется)	Включает журналирование для выбранного файла или каталога.
setfacl/getfacl	**setfacl/getfacl**	Настройка списка управления доступом для файлов и каталогов.
setfattr/getfattr	**setfattr/getfattr**	Просмотр и изменение атрибутов файла.

Полный список параметров можно найти на справочных страницах команд.

1.4.2. Другие различия

Ниже перечислены отличия в управлении GFS и GFS2.

1.4.2.1. Контекстные ссылки

GFS2 не поддерживает контекстные ссылки, использующие переменные для доступа к файлам и каталогам. Эту функциональность в GFS2 реализует команда **mount --bind** (см. Раздел 4.12, «mount --bind и контекстные ссылки»).

1.4.2.2. gfs2.ko

За реализацию файловой системы GFS отвечает модуль ядра **gfs.ko**, за реализацию GFS2 — **gfs2.ko**.

1.4.2.3. Квоты в GFS2

В GFS2 квотирование пространства по умолчанию отключено (см. Раздел 4.5, «Управление квотами в GFS2»).

1.4.2.4. Ведение журналов

GFS2 поддерживает команду **chattr**, с помощью которой можно установить флаг **j** для каталога или отдельного файла. Установка флага **+j** для каталога будет обозначать, что журналы будут вестись для всех создаваемых в нем файлов и подкаталогов. **chattr** является предпочтительным методом управления журналированием.

1.4.2.5. Динамическое добавление журналов

Журналы в GFS содержат метаданные и располагаются за пределами файловой системы, что требует увеличения размера логического тома. Журналы GFS2 представляют собой обычные файлы (хоть и скрытые), которые могут добавляться динамически при монтировании файловой системы на новых серверах (см. Раздел 4.7, «Добавление журналов»).

1.4.2.6. atime_quantum

GFS2 не поддерживает параметр **atime_quantum**, который используется в GFS для определения частоты обновлений **atime**. Вместо него GFS2 использует параметры монтирования **relatime** и **noatime**. В частности, **relatime** аналогичен параметру **atime_quantum** в GFS.

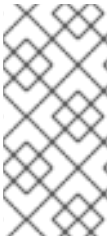
1.4.2.7. mount data=ordered/writeback

Параметр монтирования **data=ordered** означает, что измененные в ходе транзакции данные будут сохранены на диск до того, как будет сохранена сама транзакция. Это позволяет избежать появления неинициализированных блоков в случае сбоя. В режиме **data=writeback** запись данных на диск откладывается до тех пор, пока не начнется их изменение в кэше. Отложенная запись повышает скорость работы, но не защищает данные в случае внезапной остановки компьютера. По умолчанию будет выбран режим **ordered**.

1.4.2.8. gfs2_tool

Параметры **gfs2_tool** отличаются от **gfs_tool**:

- **gfs2_tool** поддерживает параметр **journals** для вывода списка настроенных журналов.
- **gfs2_tool** не поддерживает флаг **counters**, который **gfs_tool** использует для вывода статистики GFS.
- **gfs2_tool** не поддерживает флаг **inherit_jdata**. Для настройки журналирования в каталоге установите флаг **jdata** или измените его атрибуты, установив флаг **+j** с помощью **chattr**. Команда **chattr** является предпочтительным методом изменения атрибутов.



ПРИМЕЧАНИЕ

В Red Hat Enterprise Linux 6.2 команда **tunegfs2** заменила некоторые функции **gfs2_tool**. Подробную информацию можно найти на справочной странице **tunegfs2**(8). Функции **settune** и **gettune** теперь успешно выполняют параметры команды **mount**, которые могут быть определены в файле **fstab**.

1.4.2.9. gfs2_edit

Параметры **gfs2_edit** отличаются от параметров **gfs_edit**. Подробную информацию можно найти на справочных страницах **gfs2_edit** и **gfs_edit**.

1.4.3. Производительность

Некоторые функции GFS2 аналогичны GFS, но значительно улучшают производительность файловой системы.

Преимущества GFS2:

- Высокая производительность при высокой интенсивности обращения к отдельному каталогу.
- Высокая скорость синхронного ввода-вывода.
- Высокая скорость чтения данных из кэша (нет задержек блокирования).
- Высокая скорость прямого ввода-вывода для предварительно выделенных файлов при условии, что размер блоков достаточно велик (например, 4 МБ).
- Высокая скорость ввода-вывода в целом.
- **df** выполняется намного быстрее вследствие более быстрых вызовов **statfs**.
- Возможность коррекции частоты обновления **atime**.

Кроме того:

- GFS2 интегрирована в официально поддерживаемое ядро (2.6.19).
- GFS2 поддерживает:
  - расширенные атрибуты файлов (**xattr**);
  - настройку атрибутов **lsattr**() и **chattr**() с помощью **ioctl**();
  - метки времени с точностью до наносекунд.

Улучшена внутренняя эффективность GFS2:

- GFS2 экономно использует память ядра.
- В GFS2 нет необходимости в использовании счетчика генерации метаданных.

  Выделение метаданных GFS2 не требует выполнения операций чтения. Управление копиями блоков метаданных в разных журналах осуществляется за счет их отзыва из журналов перед снятием блокировки. Это предотвращает параллельное изменение

одного и того же блока в других журналах.

- GFS2 использует упрощенный менеджер журналирования (без учета несвязанных дескрипторов и изменений квот).
- **gfs2_grow** и **gfs2_jadd** используют блокирование для предотвращения одновременного выполнения нескольких экземпляров.
- Упрощен код ACL для **creat**() и **mkdir**().
- Учет несвязанных дескрипторов, обновлений квот и изменений **statfs** без необходимости повторного монтирования журнала.

ГЛАВА 2. ФУНКЦИОНАЛЬНЫЕ ОСОБЕННОСТИ И КОНФИГУРАЦИЯ GFS2

Файловая система GFS2 предназначена для взаимодействия между вычислительными узлами в кластере и организации совместного доступа к хранилищу. Механизм блокирования, реализованный на уровне TCP/IP, предотвращает потерю данных при параллельном обращении к ресурсам.

В этой главе приведены рекомендации, которые помогут улучшить производительность GFS2.

ВАЖНО

Прежде всего убедитесь, что конфигурация комплекта Red Hat High Availability соответствует вашим требованиям. При необходимости проконсультируйтесь с представителем Red Hat.

2.1. ОСОБЕННОСТИ ФОРМАТИРОВАНИЯ

Далее приведены рекомендации по форматированию GFS2.

2.1.1. Размер: чем меньше, тем лучше

В основу GFS2 положена 64-разрядная архитектура, что теоретически позволяет создать файловую систему размером до 8 эксабайт. Но на сегодняшний день максимально поддерживаемый размер составляет 100 терабайт для 64-разрядной архитектуры и 16 терабайт для 32-разрядной.

При выборе размера файловой системы руководствуйтесь принципом: чем меньше, тем лучше. Таким образом, лучше создать 10 файловых систем размером 1 ТБ, чем одну размером 10 ТБ.

Преимущества небольших файловых систем:

- Резервное копирование занимает меньше времени.
- Меньше времени уходит на проверку **`fsck.gfs2`**.
- **`fsck.gfs2`** требует меньше памяти для своей работы.

Число групп ресурсов в небольших файловых системах невелико, поэтому издержки на их поддержку тоже снижаются.

Выберите такой размер файловой системы, чтобы она не была слишком большой, но и не переполнялась.

2.1.2. Размер блоков

Начиная с Red Hat Enterprise Linux 6, команда **`mkfs.gfs2`** автоматически подбирает размер блоков исходя из топологии устройства. 4 КБ должно быть достаточно, так как размер страниц памяти в Linux по умолчанию составляет именно 4 КБ. В отличие от других файловых систем, GFS2 использует буферы ядра размером 4 КБ, поэтому ядру не приходится выполнять лишнюю работу при взаимодействии с буфером.

Обычно не рекомендуется менять стандартный размер блоков, хотя в редких случаях это допустимо — например, чтобы повысить эффективность взаимодействия с хранилищем с большим числом очень маленьких файлов.

2.1.3. Число журналов

В GFS2 на каждый узел приходится по одному журналу. Так, в кластере с 16 узлами, где только два узла монтируют файловую систему, понадобится два журнала. Если же еще один узел начнет монтировать файловую систему, журнал можно будет добавить с помощью **gfs2_jadd**.

2.1.4. Размер журналов

Размер журналов можно определить на этапе форматирования с помощью **mkfs.gfs2**. По умолчанию будет выбрано 128 МБ, что вполне достаточно для большинства приложений.

Кому-то может показаться, что 128 МБ — слишком много, и вы захотите уменьшить размер до 32 или даже до 8 МБ. И хотя такое мнение имеет право на существование, уменьшение размера журналов может отрицательно сказаться на производительности. Аналогично другим файловым системам, GFS2 осуществляет запись метаданных в журнал, прежде чем они будут записаны на диск, что позволяет их восстановить в случае внезапного отключения системы. Это быстро заполнит 8 мегабайт в журнале, вследствие чего производительность пострадает, так как файловая система должна будет ожидать выполнения записи на диск.

Итак, рекомендуемый размер журналов составляет 128 МБ. Однако для небольших файловых систем (например, размером 5 ГБ), это может оказаться непрактично, и администратор сочтет целесообразным его уменьшить, в то время как для больших файловых систем увеличение размера (например, до 256 МБ) может улучшить производительность.

2.1.5. Группы ресурсов

При форматировании **mkfs.gfs2** пространство разбивается на одинаковые секции — так называемые «группы ресурсов». Размер группы выбирается автоматически (от 32 МБ до 2 ГБ), но по желанию можно установить конкретный размер с помощью параметра **-r**.

Оптимальный размер зависит от того, насколько заполнена и как сильно фрагментирована будет файловая система.

Точный размер подбирается экспериментальным путем при тестировании кластера.

Если файловая система содержит большое число небольших групп, процесс выделения блоков будет занимать слишком много времени за счет перебора тысяч групп в ходе поиска свободных блоков. По мере заполнения файловой системы время поиска будет расти, что существенно снизит производительность.

И наоборот, наличие лишь нескольких групп большого размера может привести к частому состязанию узлов за ресурсы, что тоже негативно влияет на производительность. Так, например, если файловая система размером 10 ГБ разделена на 5 групп размером 2 ГБ, кластерные узлы будут конкурировать за доступ к ним гораздо чаще, чем в системе такого же размера, но с 320 группами по 32 МБ. Ситуация усугубляется по мере заполнения файловой системы, так как при поиске свободных блоков надо будет проверить несколько групп. Для борьбы с этой проблемой GFS2 использует два подхода:

- Заполненные группы ресурсов будут исключаться из поиска до тех пор, пока хотя бы один блок в группе не освободится. Если вы не планируете удалять файлы, степень состязания за ресурсы снизится. Однако при частом удалении и новом распределении

блоков в переполненной файловой системе это сильно снизит производительность.

- При добавлении новых блоков в файл они будут помещены в ту же группу ресурсов, где расположен файл. Таким образом, операции перехода в файле будут выполняться быстрее за счет того, что блоки будут физически расположены близко друг к другу.

При самом неблагоприятном развитии событий узлы будут непрерывно состязаться за доступ к группе ресурсов — например, если все узлы в кластере пытаются создать файлы в одном каталоге.

2.2. ФРАГМЕНТАЦИЯ

В Red Hat Enterprise Linux 6.4 процессы параллельной записи файлов были оптимизированы, что значительно уменьшило уровень фрагментации.

В GFS2 нет специальных инструментов дефрагментации, поэтому для этого придется использовать подручные средства. Фрагментированные файлы можно найти с помощью filefrag, после чего скопировать, переименовать и заменить оригинал.

2.3. ВЫДЕЛЕНИЕ БЛОКОВ

В этой секции обсуждаются вопросы размещения блоков в GFS2. Обычно при записи данных приложения не заботятся о том, как распределяются блоки, но понимание принципов, согласно которым будут выделяться блоки, поможет улучшить производительность.

2.3.1. Оставьте свободное место

По мере заполнения файловой системы найти свободные блоки становится все сложнее, и все чаще будут выделяться оставшиеся блоки, расположенные в разных местах, что приведет к значительной фрагментации. Более того, чем меньше свободных блоков, тем больше времени займет поиск из-за перебора групп ресурсов, и тем больше вероятность состязания за доступ к группам — все это отрицательно сказывается на производительности.

Исходя из вышесказанного, рекомендуется оставить хотя бы 15% свободного пространства в файловой системе (эта цифра может корректироваться в зависимости от нагрузки).

2.3.2. Выделение блоков владельцем файла

В силу особенностей менеджера распределенных блокировок (DLM, Distributed Lock Manager), степень конкуренции за доступ к ресурсам повышается, если файл создается одним узлом, а другие узлы пытаются добавить в него блоки.

Первая версия GFS использовала механизм GULM (Grand Unified Lock Manager), который управлял блокировками в кластере централизовано, что не исключало вероятность возникновения единой точки отказа. В GFS2 блокировки распределены в пределах кластера — если один узел выйдет из строя, его блокировки будут восстановлены на других узлах.

DLM использует другой подход: первый узел, заблокировавший файл, считается владельцем блокировки. Другие узлы тоже могут его заблокировать, но сначала они должны получить разрешение владельца. Каждый узел отслеживает то, какими блокировками он владеет, и каким узлам было выдано разрешение. Активация собственной блокировки происходит намного быстрее, так как для этого не требуется отдельное разрешение.

Так же как и другие файловые системы, GFS2 старается выделять файлам блоки недалеко друг от друга, чтобы сократить время перемещения записывающей головки. Если другой узел

осуществляет запись в файл, он должен будет заблокировать его группу ресурсов. Скорость работы файловой системы снизится, если дополнительно надо будет получить разрешение владельца блокировки, поэтому будет лучше, чтобы именно владелец (узел, впервые открывший файл) выделял блоки в соответствующей группе ресурсов.

2.3.3. Предварительное выделение места

Если место для файлов выделяется заранее, необходимость контроля за распределением блоков полностью отпадает, что повышает эффективность работы файловой системы. В новых версиях GFS2 эту задачу выполняет **fallocate**(1).

2.4. ПРОЕКТИРОВАНИЕ КЛАСТЕРА

При проектировании кластера следует помнить, что при увеличении числа узлов масштабирование нагрузки будет усложняться. Именно поэтому Red Hat не поддерживает GFS2 в комплексах с более чем 16 узлами.

Новую кластерную файловую систему рекомендуется поставить на испытание на 8-12 недель, чтобы убедиться в том, что производительность кластера поддерживается на должном уровне. Используйте это время для мониторинга отклонений, коррекции конфигурации и, если потребуется, проконсультируйтесь со службой поддержки Red Hat.

Прежде чем ввести кластер в эксплуатацию, следует обсудить его конфигурацию со службой поддержки Red Hat.

2.5. ОСОБЕННОСТИ ЭКСПЛУАТАЦИИ

В этой секции обсуждаются общие рекомендации по эксплуатации GFS2.

2.5.1. Noatime и nodiratime

GFS2 рекомендуется монтировать с параметрами **noatime** и **nodiratime**.

2.5.2. Размер таблиц DLM

DLM использует несколько таблиц для координации, управления и обмена информацией о блокировках. Увеличение размера таблиц может улучшить производительность. В Red Hat Enterprise Linux 6.1 стандартный размер таблиц увеличился, но по желанию его можно корректировать.

```
echo 1024 > /sys/kernel/config/dlm/cluster/lkbtbl_size
echo 1024 > /sys/kernel/config/dlm/cluster/rsbtbl_size
echo 1024 > /sys/kernel/config/dlm/cluster/dirtbl_size
```

Результаты этих команд не сохранятся после перезагрузки, поэтому поместите их в сценарий инициализации перед командами монтирования файловой системы.

Раздел 2.9, «Блокировка узлов GFS2» содержит подробную информацию.

2.5.3. Виртуальная файловая система

GFS2 может лежать в основе виртуальной файловой системы (VFS, Virtual File System). Некоторые изменения конфигурации на уровне VFS, сделанные с помощью **sysctl**(8), могут

улучшить производительность GFS2. Так, например, для просмотра текущих значений **dirty_background_ratio** и **vfs_cache_pressure** выполните:

```
sysctl -n vm.dirty_background_ratio
sysctl -n vm.vfs_cache_pressure
```

Чтобы установить новые значения:

```
sysctl -w vm.dirty_background_ratio=20
sysctl -w vm.vfs_cache_pressure=500
```

Чтобы результаты сохранялись после перезагрузки, добавьте эти выражения в **/etc/sysctl.conf**.

Оптимальные значения подбираются экспериментальным путем на стадии тестирования кластера.

2.5.4. SELinux

В целях безопасности обычно рекомендуется использовать SELinux, но в GFS2 это нецелесообразно. Дело в том, что SELinux хранит информацию об объектах файловой системы в дополнительных атрибутах, а чтение и запись таких атрибутов существенно замедляет работу GFS2. Именно поэтому SELinux в GFS2 следует отключить.

2.5.5. NFS и GFS2

Вследствие сложной организации механизма блокирования GFS2 реализация NFS поверх GFS2 требует тщательной подготовки и осторожной конфигурации. Далее обсуждаются меры предосторожности, которые надо принять при настройке доступа NFS к файловой системе GFS2.

ПРЕДУПРЕЖДЕНИЕ

Если к файловой системе настроен доступ NFS, а приложения NFS-клиента используют блокировку POSIX, то при монтировании следует добавить параметр **localflocks**. Дело в том, что попытки управления блокировками POSIX в совместно используемой файловой системе вызовут целый ряд проблем, в то время как **localflocks** позволяет рассматривать файловую систему как локальную и разрешает обработку блокировок локально на заданном узле. Для приложений NFS-клиентов подобная изоляция блокировок POSIX означает, что два клиента могут одновременно удерживать блокировку, если они монтируют ресурс с разных серверов. Если же клиенты монтируют NFS-ресурс с одного сервера, то необходимость в отдельной блокировке с каждого сервера отпадает. Если вы не уверены в необходимости добавления параметра **localflocks**, лучше его не указывать.

Ниже перечислены некоторые особенности организации NFS-доступа к GFS2.

- Red Hat поддерживает только активно-пассивную конфигурацию NFSv3 в комплексе с Red Hat High Availability со следующими характеристиками:
  - В кластере с 2-16 узлами в качестве базовой файловой системы используется GFS2.
  - Сервер NFSv3 экспортирует всю файловую систему GFS2 с одного узла.
  - При сбое NFS-сервера он будет восстановлен на другом узле в кластере (активно-пассивная конфигурация).
  - Доступ к GFS2 разрешен *только* через NFS-сервер (в том числе локальный и доступ с использованием Samba).
  - В системе отключена поддержка квот NFS.

  Это позволяет создать отказоустойчивую файловую систему и уменьшает время простоя, так как при восстановлении NFS-сервера на другом узле не требуется тратить время на проверку **`fsck`**.

- Параметр **`fsid=`** обязателен при экспорте GFS2.

- При нарушении кворума или неудачной изоляции узла в кластере доступ к логическим томам и файловой системе будет закрыт до тех пор, пока кворум не будет восстановлен. Это стоит учитывать при оценке того, поможет ли описанная выше процедура восстановления решить поставленные задачи.

2.5.6. Samba и GFS2

Начиная с Red Hat Enterprise Linux 6.2 доступ к файлам в GFS2 может быть организован при помощи Samba (SMB или Windows), что позволяет построить активную форму кластера. За информацией о кластерной конфигурации Samba обратитесь к *руководству по администрированию кластера*.

Параллельный доступ к данным на общем ресурсе Samba извне и функции аренды в кластере GFS2 не поддерживаются, что существенно замедляет работу Samba.

2.6. РЕЗЕРВНОЕ КОПИРОВАНИЕ

Резервное копирование файловой системы позволяет восстановить данные в случае их повреждения. Иногда системные администраторы полагают, что наличия зеркальных копий, снимков и RAID более чем достаточно, однако несмотря на предлагаемый ими уровень избыточности, регулярное копирование обеспечивает максимальную надежность.

Процесс резервного копирования в кластере усложняется необходимостью последовательного чтения всей файловой системы. Если операция копирования запускается с одного узла, вся информация будет храниться в его кэше до тех пор, пока другие узлы не запросят блокировку. Это отрицательно повлияет на быстродействие.

Запись содержимого кэша на диск по завершении копирования сэкономит время, которое будет затрачено другими узлами на возврат прав владения кэшем и блокировками. Это тоже не идеальный вариант, так как остальные узлы должны будут прекратить кэширование до начала резервного копирования. После завершения копирования освободите кэш:

```
echo -n 3 > /proc/sys/vm/drop_caches
```

Производительность кластера улучшится, если каждый узел будет создавать собственную резервную копию. Для этой цели можно создать сценарий с командами **rsync**.

С учетом вышесказанного, для резервного копирования лучше всего подойдет вариант с созданием аппаратного снимка в SAN с последующим копированием на другой узел. Так как резервный узел будет располагаться за пределами кластера, при монтировании снимка надо будет добавить параметры **-o lockproto=lock_nolock**.

2.7. АППАРАТНЫЕ ОСОБЕННОСТИ

Внедрение GFS2 следует проводить с учетом следующих рекомендаций по оборудованию.

- Использование высококачественных устройств хранения данных

  GFS2 можно развернуть на самых разных устройствах, но для достижения максимальной производительности рекомендуется сделать выбор в пользу качественных решений с широкими возможностями кэширования. Red Hat проводит базовое тестирование производительности SAN с Fibre Channel, но всегда можно самостоятельно проверить работоспособность выбранных решений до ввода в эксплуатацию.

- Предварительное тестирование сетевого оборудования

  Качественное оборудование является важным условием для создания надежной и эффективной файловой системы, но это не означает, что приобретение дорогостоящего оборудования является гарантией качества. Даже самые дорогие коммутаторы могут не пропускать многоадресные пакеты, передающие блокировки **fcntl**, в то время как их бюджетные аналоги могут оказаться более надежными. Окончательное решение в каждом конкретном случае стоит принимать лишь по результатам предварительного тестирования.

2.8. ПОРТАЛ ПОЛЬЗОВАТЕЛЕЙ RED HAT

Информацию о создании кластеров Red Hat Enterprise Linux на базе Red Hat GFS2 и High Availability можно найти в статье «Рекомендации по развертыванию кластера Red Hat Enterprise Linux с High Availability и GFS2» по адресу https://access.redhat.com/site/articles/40051.

2.9. БЛОКИРОВКА УЗЛОВ GFS2

Прежде чем приступить к созданию плана оптимизации производительности файловой системы, важно понимать принципы ее организации. При развертывании файловой системы на одном узле также будет создан кэш, где будут размещаться наиболее часто используемые данные. Скорость доступа к кэшу намного выше по сравнению с обращением к диску.

В GFS2 каждый узел использует собственный кэш страниц, в который подкачиваются данные с диска. Согласование кэша между узлами осуществляется с помощью механизма *glocks*, использующего функции менеджера распределенных блокировок (DLM, Distributed Lock Manager).

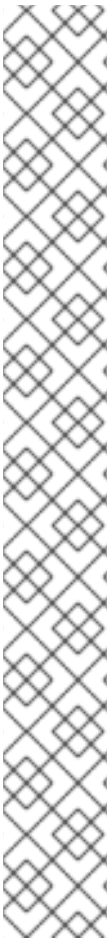
В glocks блокирование данных осуществляется на основе индексных дескрипторов — каждый дескриптор блокируется отдельно. Если блокировка была сделана в разделяемом режиме (DLM-режим: PR), заблокированные данные могут одновременно кэшироваться на других узлах, то есть они будут доступны локально на каждом узле.

Монопольный режим (DLM-режим: EX) означает, что только блокирующий узел сможет кэшировать данные. Этот режим используется операциями модификации данных, такими как **write**.

Если узел запрашивает glock, который не может быть сразу освобожден, DLM отправит блокирующим узлам запрос освобождения. По меркам быстродействия файловых систем освобождение блокировки занимает довольно много времени, в то время как освобождение совместной блокировки требует лишь сброса состояния кэша, что происходит гораздо быстрее и напрямую зависит от объема данных в кэше.

При освобождении монопольной блокировки изменения в журнале и модифицированные данные будут записаны на диск, после чего состояние кэша будет аннулировано.

Главное отличие распределенной файловой системы от обычной состоит в наличии отдельного кэша на каждом узле. Степень задержки при обращении к кэшу в обоих случаях сравнима, но задержка доступа к данным на диске значительно выше в GFS2, и эффективность падает, если другой узел уже кэшировал те же самые данные.



ПРИМЕЧАНИЕ

В силу особенностей кэширования в GFS2, для достижения наилучшей производительности следует придерживаться следующих правил:

- обращаться к inode с разных узлов только в режиме чтения или
- производить запись в inode только с одного узла.

Так как операции создания и удаления файлов из каталога вносят изменения в его дескриптор, они рассматриваются как операции записи.

В редких случаях допускается отклонение от перечисленных правил, но во избежание неоправданного снижения производительности не следует этим злоупотреблять.

Если вы используете **mmap**() для отображения файла в память с разрешениями чтения и записи, но при этом выполняются только операции чтения, в GFS2 это будет восприниматься как чтение, а в GFS — как запись. Таким образом, возможности масштабирования GFS2 на уровне ввода-вывода значительно выше по сравнению с GFS.

Если при подключении файловой системы в строке **mount** не указан параметр **noatime**, то даже операции чтения будут обновлять время доступа к файлу. Поэтому при подключении GFS2 рекомендуется отключить эту функцию, добавив **noatime**.

2.9.1. Блокировка Posix

Особенности использования блокировок Posix:

- Flocks намного эффективнее блокировок Posix.
- Программы, использующие блокировки Posix, должны избегать вызова **GETLK** в кластерных окружениях, так как это может привести к тому, что идентификатор одного и того же процесса на разных узлах будет отличаться.

2.9.2. Оптимизация производительности

В ходе оптимизации производительности программ в первую очередь следует обратить внимание на то, как они хранят данные, и можно ли это как-то улучшить.

В качестве примера рассмотрим почтовый сервер. На сервере может быть размещен один каталог spool с файлами пользователей (**mbox**) или отдельные каталоги пользователей с файлами сообщений (**maildir**). Если для доступа к почте используется протокол IMAP, эффективность работы можно повысить, связав пользователя с конкретным узлом. Таким образом, запросы чтения и удаления писем будут обслуживаться из кэша на одном узле. В случае сбоя узла сеанс можно будет начать заново на другом узле.

Если же для получения почты используется SMTP, можно настроить передачу почты пользователя конкретному узлу. Если узел не подключен, сообщение будет сохранено в spool на узле получателя. Это позволяет кэшировать файлы на одном узле, но в то же время разрешает прямой доступ к почте в случае его сбоя.

Такая организация позволяет эффективно использовать кэш страниц и делает сбои прозрачными для **imap** и **smtp**.

Отдельно можно оптимизировать процесс резервного копирования, по возможности создавая копии данных напрямую с кэширующего узла. Если сценарий копирования запускается регулярно в одно и то же время, и в это же время регистрируется задержка ответа приложений, это может служить знаком неэффективного использования кэша страниц.

Если вы можете остановить работу приложения, чтобы беспрепятственно завершить резервное копирование, отлично — вас эта проблема не коснется. Но если такой возможности нет, и копирование запущено с одного узла, то после его окончания в его кэше будет находиться значительная часть содержимого файловой системы. Это существенно замедлит время ответа этого узла при обращении к нему с других узлов. Последствия можно смягчить, очистив кэш:

```
echo -n 3 >/proc/sys/vm/drop_caches
```

Но это тоже не является идеальным вариантом — лучше разрешить совместный доступ к рабочим данным в режиме чтения или обращаться к ним с одного узла.

2.9.3. Статистика блокировок

Неэффективное кэширование в GFS2 может привести к задержкам при обработке ввода-вывода и снизить производительность кластера в целом. Для определения причины такого поведения можно проверить статистику блокировок.

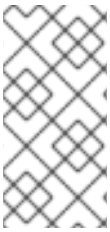
Приложение C, *Мониторинг событий и файл glocks* подробно обсуждает тему анализа статистики блокировок.

Статистику можно собрать при помощи **debugfs**. Если **debugfs** подключена в **/sys/kernel/debug/**, путь будет выглядеть так:

```
/sys/kernel/debug/gfs2/ФС/glocks
```

Это обычный текстовый файл. Секции блокировок начинаются с «G:». Внутри секции строки начинаются с пробела и содержат подробную информацию о блокировке.

Чтобы сохранить сведения о блокировках работающей программы, используйте **`cat`** для вывода содержимого этого файла. В зависимости от числа кэшируемых дескрипторов и объема ОЗУ продолжительность этой операции может быть разной.



ПРИМЕЧАНИЕ

Можно создать две копии файла **`glocks`** — одну сразу, вторую через пару минут — и сравнить состояние блокировок. Если вы обнаружили блокировку, состояние которой не изменилось, сообщите об этом в службу поддержки Red Hat, так как это служит признаком ошибки.

Строки, начинающиеся с «H:» (holders), обозначают ожидающие или уже активные запросы блокировки. Значение «W» в поле «f:» (flags) означает ожидающий запрос, а «H» — активную блокировку. Обратите внимание на число ожидающих запросов: слишком большое число означает, что запросы обрабатываются слишком медленно.

Таблица 2.1, «Флаги glock» и Таблица 2.2, «Флаги удерживания glock» содержат описание флагов.

Таблица 2.1. Флаги glock

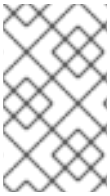
Флаг	Значение	Описание
b	Blocking	Используется вместе с флагом «l» (Locked) и означает, что запрашиваемая операция DLM может привести к блокированию. Этот флаг снимается для операций снижения режима блокирования и пробных блокировок. Его главная цель — сбор статистики времени ответа DLM без учета того, сколько времени займет снижение режима блокирования.
d	Pending demote	Получен запрос снижения режима блокировки, но glock уже удерживается, и время минимального обслуживания не истекло.
D	Demote	Есть запрос снижения режима блокирования (локальный или удаленный).
f	Log flush	Прежде чем освободить glock, необходимо сохранить журнал.
F	Frozen	Идет восстановление — ответы удаленных узлов игнорируются.
i	Invalidate in progress	Сброс страниц кэша под блокировкой.
I	Initial	Выбранному glock назначена блокировка DLM.
l	Locked	Glock в процессе изменения состояния.
L	LRU	Glock в списке LRU.

Флаг	Значение	Описание
o	Object	Означает, что glock связан с объектом. Объекты могут быть разных типов. Так, тип 2 означает дескриптор, 3 — группу ресурсов.
p	Demote in progress	Получен запрос понижения режима блокировки.
q	Queued	Устанавливается при наличии ожидающих запросов и снимается, если запросов нет. Используется алгоритмом расчета минимального времени удерживания glock.
r	Reply pending	Ответ, полученный от удаленного узла, ожидает обработки.
y	Dirty	Прежде чем освободить glock, необходимо сохранить данные.

Таблица 2.2. Флаги удерживания glock

Флаг	Значение	Описание
a	Async	Не ждать результата glock (будет запрошен позднее).
A	Any	Принимает любой совместимый режим блокировки.
c	No cache	Если не заблокировано, сразу снизить режим блокировки DLM.
e	No expire	Игнорирует последующие запросы снятия блокировки.
E	exact	Требуется конкретный режим блокировки.
F	First	Первый запрос, захвативший блокировку.
H	Holder	Блокировка установлена.
p	Priority	Ставит запрос блокировки в начало очереди.
t	Try	Пробная блокировка.
T	Try 1CB	Пробная блокировка с ответом.
W	Wait	Ожидание обработки запроса.

Определив, какая именно блокировка является причиной конфликта, в строке «G:» найдите «n:*тип*/*номер*». Если тип равен 2, за ним будет следовать номер дескриптора в шестнадцатеричном формате. Чтобы определить файл, которому принадлежит дескриптор, преобразуйте номер в десятичный формат и выполните: **`find -inum` *`десятичное_число`***.



ПРИМЕЧАНИЕ

Вызов **find** в файловой системе с высокой степенью конкуренции за блокировки может усугубить ситуацию, поэтому прежде чем начать поиск спорных дескрипторов, рекомендуется остановить приложение.

Ниже приведено описание типов glock.

Таблица 2.3. Типы glock

Номер	Тип	Описание
1	Trans	Блокировка транзакции
2	Inode	Индексный дескриптор
3	Rgrp	Метаданные группы ресурсов
4	Meta	Суперблок
5	Iopen	Идентифицирует открытый дескриптор
6	Flock	Вызов **flock**(2)
8	Quota	Операции с квотой
9	Journal	Мьютекс журнала

Другой распространенный тип glock, за который могут конкурировать процессы, — это тип 3 (группа ресурсов). В остальных случаях, если вы заметили слишком частые запросы других типов glock при стандартной нагрузке, следует обратиться в службу поддержки Red Hat.

Большое число запросов, ожидающих третий тип блокирования, может быть вызвано тем, что число узлов в кластере значительно превышает число групп ресурсов (что приводит к частому обращению разных узлов к одной секции диска) или обычным переполнением файловой системы. В обоих случаях проблема решается увеличением пространства и наращиванием файловой системы (с помощью **gfs2_grow**).

ГЛАВА 3. НАЧАЛО РАБОТЫ

В этой главе рассказывается об исходной конфигурации GFS2.

- Раздел 3.1, «Предварительные требования»
- Раздел 3.2, «Первоначальная конфигурация»

3.1. ПРЕДВАРИТЕЛЬНЫЕ ТРЕБОВАНИЯ

Подготовьте системы к установке файловой системы:

- Раздел 1.2, «Проектирование структуры GFS2» содержит обзор ключевых характеристик файловой системы.
- Системные часы узлов GFS2 должны быть синхронизированы, для чего рекомендуется использовать NTP (Network Time Protocol).

ПРИМЕЧАНИЕ

Системное время на узлах не должно сильно расходиться во избежание ненужного обновления меток времени inode, так как это снизит производительность кластера.

- Чтобы установить GFS2 в кластерном окружении, необходимо настроить набор расширений CLVM. В свою очередь, для работы CLVM необходимо, чтобы в системе работал комплект Red Hat Cluster Suite и служба **`clvmd`**. Информацию о CLVM можно найти в руководстве LVM, а о Red Hat Cluster Suite — в руководстве по администрированию кластера.

3.2. ПЕРВОНАЧАЛЬНАЯ КОНФИГУРАЦИЯ

Основные этапы исходной настройки GFS2:

1. Создание логических томов.
2. Создание файловой системы.
3. Монтирование файловой системы.

Далее эти этапы рассматриваются более подробно.

1. С помощью LVM создайте по одному логическому тому для каждой файловой системы GFS2.

ПРИМЕЧАНИЕ

Для автоматизации процесса добавления логических томов можно использовать сценарии **`init.d`** в Red Hat Cluster Suite. *Настройка и управление кластером Red Hat* содержит подробную информацию об этих сценариях.

2. Создайте файловые системы, присвоив им уникальные имена (см. Раздел 4.1, «Создание файловой системы»).

   Команды создания GFS2:

   ```
   mkfs.gfs2 -p lock_dlm -t кластер:ФС -j число_журналов устройство
   ```

   ```
   mkfs -t gfs2 -p lock_dlm -t таблица_блокирования -j число_журналов
   устройство
   ```

   Раздел 4.1, «Создание файловой системы» содержит подробную информацию.

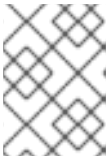
3. Смонтируйте GFS2 на всех узлах (см. Раздел 4.2, «Монтирование файловой системы»).

   Формат команды:

   mount *устройство каталог*

   mount -o acl *устройство каталог*

   Параметр **-o acl** позволяет управлять списками ACL. Если не указан, пользователи будут иметь возможность просмотра списков ACL (**getfacl**), но не смогут их изменить (**setfacl**).

   ПРИМЕЧАНИЕ

   Подключение и отключение файловой системы можно автоматизировать с помощью сценариев **init.d**.

ГЛАВА 4. УПРАВЛЕНИЕ GFS2

Содержание главы:

- Раздел 4.1, «Создание файловой системы»
- Раздел 4.2, «Монтирование файловой системы»
- Раздел 4.3, «Отключение файловой системы»
- Раздел 4.5, «Управление квотами в GFS2»
- Раздел 4.6, «Увеличение размера файловой системы»
- Раздел 4.7, «Добавление журналов»
- Раздел 4.8, «Журналирование данных»
- Раздел 4.9, «Обновление **atime**»
- Раздел 4.10, «Временная остановка файловой системы»
- Раздел 4.11, «Проверка файловой системы»
- Раздел 4.12, «mount --bind и контекстные ссылки»
- Раздел 4.13, «mount --bind и порядок подключения»
- Раздел 4.14, «Функции отзыва GFS2»

4.1. СОЗДАНИЕ ФАЙЛОВОЙ СИСТЕМЫ

GFS2 можно создать при помощи **mkfs.gfs2** или **mkfs -t gfs2**. Файловая система будет создана в пределах активного логического тома. Для вызова **mkfs.gfs2** потребуются следующие исходные данные:

- протокол блокирования или имя модуля (для кластера используется **lock_dlm**);
- имя кластера (если файловая система создается в кластере);
- число журналов (по одному журналу на узел, с которого будет монтироваться файловая система).

В строке команды **mkfs** надо добавить параметр ***-t gfs2***, чтобы определить тип файловой системы.

ПРИМЕЧАНИЕ

Если файловая система создается средствами **mkfs.gfs2**, ее размер нельзя будет уменьшить, но можно будет увеличить (см. Раздел 4.6, «Увеличение размера файловой системы»).

4.1.1. Формат команд

Формат команд создания кластерной GFS2:

```
mkfs.gfs2 -p протокол -t таблица_блокирования -j число_журналов устройство
```

```
mkfs -t gfs2 -p протокол -t таблица_блокирования -j число_журналов
устройство
```

Ниже приведены команды создания локальной файловой системы GFS2.

ПРИМЕЧАНИЕ

В Red Hat Enterprise Linux 6 использование GFS2 как локальной файловой системы на одном узле не поддерживается.

```
mkfs.gfs2 -p протокол -j число_журналов устройство
```

```
mkfs -t gfs2 -p протокол -j число_журналов устройство
```

ПРЕДУПРЕЖДЕНИЕ

Неверное определение параметров ***протокол*** и ***таблица_блокирования*** может привести к повреждению файловой системы и lockspace.

протокол

Имя блокирующего протокола. Для кластера используется **lock_dlm**.

таблица_блокирования

Этот параметр используется при конфигурации кластера и определяется в формате ***кластер:ФС***.

- ***кластер*** — имя кластера, в котором создается файловая система.
- ***ФС*** — имя файловой системы длиной от 1 до 16 символов, которое должно быть уникальным и не повторять имена других файловых систем в кластере под управлением **lock_dlm** и файловых систем на каждом узле (**lock_dlm**, **lock_nolock**).

число_журналов

Для каждого узла, подключающего файловую систему, необходим один журнал. Если число не определено, будет создан всего один журнал. Дополнительные журналы можно будет добавить позднее (см. Раздел 4.7, «Добавление журналов»).

устройство

Логический или физический том.

4.1.2. Примеры

В приведенном ниже примере на устройстве **/dev/vg01/lvol0** в кластере **alpha** будет создана файловая система **mydata1** с восемью журналами.

```
mkfs.gfs2 -p lock_dlm -t alpha:mydata1 -j 8 /dev/vg01/lvol0
```

```
mkfs -t gfs2 -p lock_dlm -t alpha:mydata1 -j 8 /dev/vg01/lvol0
```

Аналогичным образом создадим файловую систему **mydata2** на **/dev/vg01/lvol1**.

```
mkfs.gfs2 -p lock_dlm -t alpha:mydata2 -j 8 /dev/vg01/lvol1
```

```
mkfs -t gfs2 -p lock_dlm -t alpha:mydata2 -j 8 /dev/vg01/lvol1
```

4.1.3. Полный список параметров

Таблица 4.1, «Параметры **mkfs.gfs2**» содержит полный перечень параметров **mkfs.gfs2**.

Таблица 4.1. Параметры mkfs.gfs2

Флаг	Параметр	Описание
-c	***МБ***	Размер файла изменений квот журнала в ***мегабайтах***.
-D		Вывод отладочной информации.
-h		Справка с описанием параметров команды.
-J	***МБ***	Размер журнала в мегабайтах. По умолчанию используется 128 МБ, минимальный размер — 8 МБ. Большие журналы улучшают производительность, но используют больше памяти.
-j	***число_журналов***	Число журналов. Для каждого узла, монтирующего файловую систему, необходимо создать один журнал. По умолчанию создается один журнал. Дополнительные журналы могут быть добавлены позднее без необходимости наращивания файловой системы.
-O		Не запрашивать подтверждение перед записью файловой системы.

Флаг	Параметр	Описание
-p	***протокол***	Имя блокирующего протокола. Допустимые значения: **lock_dlm** — стандартный модуль блокировки в кластерной файловой системе. **lock_nolock** — применяется, если GFS2 создается как локальная файловая система на одном узле.
-q		Подавляет вывод на экран.
-r	***МБ***	Размер ресурсных групп в мегабайтах (по умолчанию 32 МБ). Максимальный размер — 2048 МБ. Большой размер может снизить производительность в больших файловых системах. Если параметр не задан, mkfs.gfs2 сделает выбор исходя из размера файловой системы — например, 256 МБ для среднего размера.
-t	***таблица_блокирования***	Уникальный идентификатор, определяющий поле в таблице блокирования. Используется только протоколом **lock_dlm**. Определяется в формате ***Кластер:ФС***. ***Кластер*** — имя кластера, где создается файловая система. Она будет доступна только узлам в пределах этого кластера. Имя кластера определяется с помощью утилиты конфигурации кластера и хранится в **/etc/cluster/cluster.conf**. ***ФС*** — имя файловой системы длиной от 1 до 16 символов, которое должно быть уникальным в пределах кластера.
-u	***МБ***	Исходный размер файла тегов.
-V		Возвращает версию команды.

4.2. МОНТИРОВАНИЕ ФАЙЛОВОЙ СИСТЕМЫ

Прежде чем подключить GFS2, убедитесь, что она расположена в пределах активного тома, и все необходимые кластерные и блокирующие службы выполняются (см. руководство по настройке и управлению кластером Red Hat).

ПРИМЕЧАНИЕ

Если **cman** не был запущен, попытка монтирования завершится ошибкой:

```
[root@gfs-a24c-01 ~]# mount -t gfs2 -o noatime
/dev/mapper/mpathap1 /mnt
gfs_controld join connect error: Connection refused
error mounting lockproto lock_dlm
```

Чтобы включить функции списков управления доступом, добавьте параметр **-o acl**, в противном случае пользователи будут иметь возможность просмотра списков ACL (**getfacl**), но не смогут их изменять (**setfacl**).

4.2.1. Формат команд

Подключение без возможности изменения ACL

```
mount устройство точка_монтирования
```

Подключение с возможностью изменения ACL

```
mount -o acl устройство точка_монтирования
```

-o acl

Параметр для манипулирования списками ACL.

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог, в который монтируется GFS2.

4.2.2. Пример

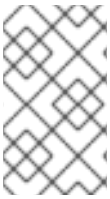
В этом примере файловая система на **/dev/vg01/lvol0** будет подключена в **/mygfs2**.

```
mount /dev/vg01/lvol0 /mygfs2
```

4.2.3. Полная форма

```
mount устройство точка_монтирования -o параметр
```

-o параметр принимает стандартные значения **mount -o** и перечисленные ниже значения (см. Таблица 4.2, «Параметры монтирования GFS2»). Несколько параметров разделяются запятой без пробела.



ПРИМЕЧАНИЕ

Параметры монтирования GFS2 могут использоваться совместно со стандартными параметрами **mount**. Подробную информацию можно найти на справочной странице **mount**.

Таблица 4.2, «Параметры монтирования GFS2» содержит полный список значений для передачи аргументу **-o**.

ПРИМЕЧАНИЕ

Таблица включает описание параметров для локальных файловых систем. Начиная с Red Hat Enterprise Linux 6 использование GFS2 в индивидуальных системах ограничивается подключением снимков кластерных файловых систем с целью резервного копирования.

Таблица 4.2. Параметры монтирования GFS2

Параметр	Описание
acl	Разрешает управлять списками ACL. Если файловая система подключена без **acl**, пользователи будут иметь возможность просмотра списков ACL (**getfacl**), но не смогут их изменять (**setfacl**).
data=[ordered|writeback]	**data=ordered** приведет к переносу измененных в ходе транзакции данных на диск до сохранения транзакции в журнал. В случае сбоя это позволит избежать появления в файле неинициализированных блоков. В режиме **data=writeback** запись данных на диск откладывается до тех пор, пока не начнется их изменение в кэше. Отложенная запись повышает скорость работы, но не защищает данные в случае внезапной остановки компьютера. По умолчанию будет выбран режим **ordered**.
ignore_local_fs **Внимание!** Эта опция *НЕ* должна использоваться, если к файловым системам GFS2 открыт общий доступ.	Принудительное использование GFS2 как распределенной файловой системы . Если используется **lock_nolock**, флаг **localflocks** будет установлен автоматически.

Параметр	Описание
localflocks **Внимание!** Эта опция *НЕ* должна использоваться, если к файловым системам GFS2 открыт совместный доступ.	Разрешает выполнять flock и fcntl на уровне VFS (Virtual File System). Параметр **lock_nolock** автоматически установит флаг **localflocks**.
lockproto=*протокол*	Протокол блокирования. Если не указан, имя протокола будет прочитано из суперблока файловой системы.
locktable=*таблица*	Таблица блокирования.
quota=[off/account/on]	Управление квотами. Значение **account** включит статистику использования пространства для каждого UID/GID. Предельные значения не будут принуждаться. По умолчанию **quota=off**.
errors=panic|withdraw	**errors=panic** означает, что ошибки файловой системы вызовут панику ядра. По умолчанию используется **errors=withdraw**, что отключает файловую систему до следующей перезагрузки (см. Раздел 4.14, «Функции отзыва GFS2»).
discard/nodiscard	Генерация запросов «discard» для освобожденных блоков.
barrier/nobarrier	Заставляет GFS2 отправлять барьерные запросы при записи журнала. По умолчанию эта функция включена, но будет отключена, если устройство не поддерживает барьеры ввода-вывода. Настоятельно рекомендуется использовать барьеры в GFS2 за исключением случаев, когда блочное устройство по умолчанию не может потерять содержимое кэша (например, если устройство подключено к UPS или не имеет кэша записи).
quota_quantum=*сек.*	Интервал синхронизации файла квот в секундах (по умолчанию 60 секунд). Большие значения могут улучшить эффективность операций с квотами, так как они будут вызываться реже, в то время как меньшие значения снижают риск превышения квоты.
statfs_quantum=*сек.*	Интервал синхронизации изменений в главном файле **statfs** (по умолчанию 30 секунд). Если равен нулю, **statfs** будет содержать наиболее актуальные значения.
statfs_percent=*число*	Процент изменений **statfs**, по достижении которого изменения будут сохранены в главный файл **statfs**. Игнорируется, если **statfs_quantum=0**.

4.3. ОТКЛЮЧЕНИЕ ФАЙЛОВОЙ СИСТЕМЫ

Отключение GFS2 осуществляется при помощи **umount**.

ПРИМЕЧАНИЕ

Информацию о **umount** можно найти на справочной странице.

4.3.1. Формат команд

```
umount точка_монтирования
```

точка_монтирования

Каталог, в который смонтирована файловая система.

4.4. ОСОБЕННОСТИ ПОДКЛЮЧЕНИЯ GFS2

Если файловая система была подключена вручную, а не в **fstab**, то при выключении компьютера она не будет отключена вместе с другими файловыми системами в ходе выполнения сценария отключения GFS2. После этого будет запущен стандартный сценарий завершения работы, который остановит пользовательские процессы, в том числе кластерную инфраструктуру, и попытается отключить файловую систему, что приведет к зависанию системы, так как кластерная инфраструктура к этому моменту уже будет остановлена.

Чтобы это предотвратить, необходимо:

- добавить запись монтирования GFS2 в файл **fstab** для ее автоматического монтирования во время запуска системы;
- если GFS2 была подключена вручную с помощью **mount**, прежде чем перезагрузить или выключить компьютер, отключите ее при помощи **umount**.

Если система зависла в процессе **umount**, попробуйте принудительно ее перезагрузить. Так как синхронизация данных к этому моменту завершена, риска потери данных нет.

4.5. УПРАВЛЕНИЕ КВОТАМИ В GFS2

Квоты в файловой системе предназначены для ограничения пространства пользователей. Изначально ограничений нет. Если во время монтирования указан параметр **quota=on** или **quota=account**, GFS2 будет отслеживать занятое пространство каждого пользователя, даже если ограничения не заданы, и периодически обновляет данные квот, чтобы в случае сбоя системы не было необходимости в восстановлении статистики использования пространства.

Узел GFS2 обновляет файл квот через определенные интервалы, чтобы не замедлять работу. Недостаток такого подхода заключается в том, что это может привести к превышению установленного лимита между обновлениями. Минимизировать риск можно, сокращая интервал синхронизации по мере приближения к лимиту.

ПРИМЕЧАНИЕ

Начиная с Red Hat Enterprise Linux 6.1, GFS2 поддерживает стандартную функциональность квот Linux. Для этого потребуется установить пакет **quota**. Далее подразумевается, что управление квотами осуществляется с помощью этих инструментов.

Управление квотами в предыдущих выпусках Red Hat Enterprise Linux осуществляется средствами **gfs2_quota** (см. Приложение A, ***gfs2_quota***).

4.5.1. Настройка дисковых квот

Для добавления дисковых квот надо выполнить следующее:

1. Настроить квоты в строгом режиме или включить режим статистики.
2. Инициализировать базу данных квот, добавив актуальную статистику использования блоков.
3. Определить правила квот (в режиме account правила игнорируются).

Далее эти этапы будут рассмотрены более подробно.

4.5.1.1. Настройка квот

В GFS2 квоты по умолчанию отключены. Чтобы их включить, во время монтирования добавьте параметр **quota=on**.

quota=account позволяет использовать функции квот и следить за использованием пространства, но без принуждения жесткого лимита.

4.5.1.1.1. Формат команд

Чтобы включить квоты, при подключении файловой системы укажите параметр **quota=on**.

```
mount -o quota=on устройство точка_монтирования
```

Подключение с активацией режима статистики квот:

```
mount -o quota=account устройство точка_монтирования
```

Чтобы отключить квоты, укажите **quota=off**. Этот вариант используется по умолчанию.

```
mount -o quota=off устройство точка_монтирования
```

quota={on|off|account}

on — включает использование квот;

off — отключает использование квот;

account — статистика использования пространства.

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог, в который монтируется GFS2.

4.5.1.1.2. Примеры

В следующем примере файловая система на **/dev/vg01/lvol0** подключается в **/mygfs2** с активацией квот:

```
mount -o quota=on /dev/vg01/lvol0 /mygfs2
```

Ниже **/dev/vg01/lvol0** подключается в **/mygfs2**. Использование квот включено в режиме статистики.

```
mount -o quota=account /dev/vg01/lvol0 /mygfs2
```

4.5.1.2. Создание базы данных квот

Прежде чем включить квотирование, необходимо выполнить проверку **quotacheck**.

quotacheck проверит файловые системы с квотами и создаст таблицу с информацией о занятом пространстве, откуда операционная система и будет получать данные.

Аргументы **-u** и **-g** отвечают за инициализацию квот для пользователей и групп соответственно. Так, чтобы создать исходные файлы квот для файловой системы в **/home**, выполните:

```
quotacheck -ug /home
```

4.5.1.3. Установка квот для пользователей

Команда **edquota** установит исходные лимиты для пользователей. Если файловая система смонтирована с параметром **quota=account**, использование пространства в пределах выделенных квот будет отслеживаться, но не будет ограничиваться.

В режиме root выполните:

```
edquota пользователь
```

Повторите эту команду для всех пользователей, для которых надо установить квоты. Например, если в **/etc/fstab** включено квотирование для раздела **/home** на **/dev/VolGroup00/LogVol02**, то результат команды **edquota testuser** будет выглядеть так:

```
Disk quotas for user testuser (uid 501):
Filesystem                blocks     soft     hard    inodes   soft
hard
/dev/VolGroup00/LogVol02  440436        0        0
```

ПРИМЕЧАНИЕ

Выбор текстового редактора, используемого командой **edquota**, определяется значением **EDITOR** в **~/.bash_profile**.

Первый столбец содержит имя файловой системы, второй — число занятых блоков. Следующие столбцы определяют лимиты.

Гибкий лимит определяет максимальный размер пространства, который можно временно обойти.

Жесткий лимит определяет максимально доступный размер пространства, который не может быть превышен ни при каких обстоятельствах.

GFS2 не поддерживает квотирование индексных дескрипторов, поэтому для GFS2 эти столбцы останутся пустыми.

Нулевой лимит снимает ограничения. Отредактируйте значения, чтобы установить собственные ограничения. Пример:

```
Disk quotas for user testuser (uid 501):
Filesystem                blocks     soft     hard    inodes   soft
hard
/dev/VolGroup00/LogVol02  440436   500000   550000
```

Результат изменения квот можно проверить:

```
quota testuser
```

4.5.1.4. Установка квот для групп пользователей

Квоты можно установить не только для отдельных пользователей, но и для целых групп. Чтобы включить квотирование, при монтировании файловой системы добавьте параметр **account=on**.

Пример настройки квот для группы **devel**:

```
edquota -g devel
```

Квоты будут инициализированы:

```
Disk quotas for group devel (gid 505):
Filesystem                blocks    soft     hard    inodes   soft   hard
/dev/VolGroup00/LogVol02  440400       0        0
```

GFS2 не поддерживает квотирование на уровне inode, поэтому для GFS2 эти столбцы останутся пустыми. Измените значения лимитов по своему усмотрению и сохраните файл.

Результат настройки квот можно проверить:

```
quota -g devel
```

4.5.2. Repquota

Администратор может следить за тем, как пользователи используют доступное им пространство.

Если пользователи регулярно превышают установленный лимит, администратор может рекомендовать более эффективно использовать пространство или увеличить квоту.

repquota создает отчет об использовании пространства. Вывод **repquota /home** может выглядеть так:

```
*** Report for user quotas on device /dev/mapper/VolGroup00-LogVol02
Block grace time: 7days; Inode grace time: 7days
   Block limits   File limits
User  used soft hard grace used soft hard grace
----------------------------------------------------------------------
root      --      36       0       0              4     0     0
kristin   --     540       0       0            125     0     0
testuser  --  440400  500000  550000          37418     0     0
```

Просмотр статистики по всем квотируемым файловым системам:

```
repquota -a
```

Два минуса после имени пользователя означают, что квота была превышена. Если был достигнут гибкий предел, вместо первого минуса будет показан плюс. Второй минус служит индикатором лимита inode, но так как GFS2 не поддерживает квотирование inode, он не меняется. Столбец **grace** останется пустым, так как GFS2 не поддерживает период ожидания.

Выполнение **repquota** в NFS не поддерживается, независимо от типа локальной файловой системы.

4.5.3. Quotacheck

Если квотирование было отключено на протяжении длительного периода времени, то прежде чем его включить, рекомендуется выполнить проверку **quotacheck**. Команда **quotacheck** создаст, проверит и обновит файлы квот, что поможет подтвердить их актуальность, особенно если при сбое системы файловая система была внезапно отключена.

Подробную информацию можно найти на справочной странице **quotacheck**.

ПРИМЕЧАНИЕ

Проверку **quotacheck** рекомендуется выполнять в периоды относительного бездействия файловой системы.

4.5.4. Quotasync

GFS2 хранит информацию о квотах в файле, который обновляется каждые 60 секунд, а не после каждой записи в файловую систему, что позволяет избежать конфликтов при одновременном изменении файла разными узлами.

По мере приближения к лимиту частота обновления файла будет динамически сокращаться, что уменьшает риск перехода границы. Исходный интервал синхронизации определяется параметром **quota_quantum** и по умолчанию равен 60 секундам (см. Таблица 4.2, «Параметры

монтирования GFS2»). Значение **quota_quantum** сбрасывается после отключения файловой системы, но его можно настроить вручную при монтировании при помощи **mount -o remount**.

С помощью **quotasync** также можно синхронизировать файлы квот в кластере в любое время между автоматическими обновлениями GFS2.

4.5.4.1. Формат команд

Синхронизация квот

```
quotasync [-ug] -a|точка_монтирования...
```

u

Синхронизация квот пользователей.

g

Синхронизация квот групп.

a

Синхронизация всех файловых систем, использующих квоты. Если параметр не задан, необходимо указать точку монтирования файловой системы.

точка_монтирования

Каталог подключения файловой системы.

Коррекция интервала синхронизации

```
mount -o quota_quantum=интервал,remount устройство точка_монтирования
```

точка_монтирования

Каталог подключения файловой системы.

интервал

Интервал синхронизации в секундах. Чем меньше значение, тем лучше точность, но тем больше это будет влиять на быстродействие.

4.5.4.2. Примеры

Пример синхронизации квот в файловой системе **/mnt/mygfs2**:

```
# quotasync -ug /mnt/mygfs2
```

Далее интервал синхронизации квот файловой системы **/mnt/mygfs2** в **/dev/volgroup/logical_volume** будет увеличен до 1 часа (3600 секунд).

```
# mount -o quota_quantum=3600,remount /dev/volgroup/logical_volume
/mnt/mygfs2
```

4.5.5. Дополнительные ресурсы

Информацию о квотах можно найти на справочных страницах:

- **quotacheck**,
- **edquota**,
- **repquota**,
- **quota**.

4.6. УВЕЛИЧЕНИЕ РАЗМЕРА ФАЙЛОВОЙ СИСТЕМЫ

gfs2_grow позволяет расширить файловую систему. Обычно это делается после увеличения размера устройства, где она расположена. Выполнение **gfs2_grow** приведет к заполнению свободного пространства между текущей границей файловой системы и новой границей устройства. После завершения будет обновлен индекс ресурсов файловой системы, и уже после этого кластер сможет утилизировать новое пространство.

gfs2_grow запускается в подключенной файловой системе, но только на одном узле в кластере. Остальные узлы смогут использовать новое пространство автоматически.

ПРИМЕЧАНИЕ

Если файловая система была создана с помощью **mkfs.gfs2**, ее нельзя будет уменьшить.

4.6.1. Формат команд

```
gfs2_grow точка_монтирования
```

точка_монтирования

Каталог подключения файловой системы.

4.6.2. Комментарии

Перед выполнением **gfs2_grow**:

- Создайте резервную копию важных данных.
- Определите том, где расположена файловая система. Для этого выполните **df** ***точка_монтирования***.
- С помощью LVM увеличьте размер тома. Руководство администратора LVM содержит информацию об управлении томами LVM.

После **gfs2_grow** выполните **df** для проверки доступного пространства.

4.6.3. Примеры

Пример увеличения файловой системы в **/mygfs2fs**.

```
[root@dash-01 ~]# gfs2_grow /mygfs2fs
FS: Mount Point: /mygfs2fs
FS: Device:      /dev/mapper/gfs2testvg-gfs2testlv
FS: Size:        524288 (0x80000)
FS: RG size:     65533 (0xfffd)
DEV: Size:       655360 (0xa0000)
The file system grew by 512MB.
gfs2_grow complete.
```

4.6.4. Полная форма

```
gfs2_grow [параметры] {точка_монтирования | устройство}
[точка_монтирования | устройство]
```

точка_монтирования

Каталог подключения файловой системы.

устройство

Устройство, на котором расположена файловая система.

Таблица 4.3, «Параметры gfs2_grow » содержит список доступных параметров.

Таблица 4.3. Параметры gfs2_grow

Параметр	Описание
-h	Краткая справка.
-q	Отключает подробный вывод.
-r МБ	Размер новой группы ресурсов в мегабайтах (по умолчанию 256 МБ).
-T	Тестовый режим. Выполняет все вычисления, но не записывает данные на диск.
-V	Возвращает версию команды.

4.7. ДОБАВЛЕНИЕ ЖУРНАЛОВ

В GFS2 журналы можно добавить в любое время с помощью **gfs2_jadd**. Команда запускается в подключенной файловой системе на одном узле — изменения автоматически отразятся на других узлах.

ПРИМЕЧАНИЕ

Если файловая система переполнена, **gfs2_jadd** завершится неудачей, даже если логический том в ее основе был увеличен. Дело в том, что журналы хранятся в обычных файлах, а не в виде метаданных, поэтому увеличения нижележащего логического тома будет недостаточно, и надо будет нарастить саму файловую систему.

Определите текущее число журналов, выполнив **gfs2_tool** с аргументом **journals**:

```
[root@roth-01 ../cluster/gfs2]# gfs2_tool journals /mnt/gfs2
journal2 - 128MB
journal1 - 128MB
journal0 - 128MB
3 journal(s) found.
```

4.7.1. Формат команд

```
gfs2_jadd -j число точка_монтирования
```

число_журналов

Число журналов для добавления.

точка_монтирования

Каталог подключения файловой системы.

4.7.2. Примеры

Добавление одного журнала для **/mygfs2**:

```
gfs2_jadd -j1 /mygfs2
```

Добавление двух журналов для **/mygfs2**:

```
gfs2_jadd -j2 /mygfs2
```

4.7.3. Полная форма

```
gfs2_jadd [параметры] {точка_монтирования | устройство}
[точка_монтирования | устройство]
```

точка_монтирования

Каталог подключения файловой системы.

устройство

Устройство, на котором расположена файловая система.

Таблица 4.4, «Параметры gfs2_jadd» содержит полный список параметров.

Таблица 4.4. Параметры gfs2_jadd

Флаг	Параметр	Описание
-h		Краткая справка.
-J	***МБ***	Размер журнала в мегабайтах. По умолчанию используется 128 МБ, при этом минимально допустимый размер — 32 МБ. Чтобы добавить журналы разных размеров, надо будет выполнить **gfs2_jadd** несколько раз. Указанная величина округляется в меньшую сторону так, чтобы она была кратной размеру сегмента журнала, определенного при создании файловой системы.
-j	***число_журналов***	Число журналов. По умолчанию будет добавлен один журнал.
-q		Отключает подробный вывод.
-V		Возвращает версию команды.

4.8. ЖУРНАЛИРОВАНИЕ ДАННЫХ

Обычно GFS2 сохраняет в журнал только метаданные, записывая изменения на диск в процессе синхронизации. Вызов **fsync()** для файла приведет к немедленной записи файла на диск.

Для очень маленьких файлов сохранение данных в журнал может быть эффективным, так как это сокращает их время обработки процессом **fsync()**. Однако по мере увеличения размера файлов скорость записи снижается, нивелируя это преимущество.

Журналирование данных может повысить производительность приложений, синхронизация которых осуществляется с помощью **fsync()**.

Включение журналирования каталога автоматически включит его для создаваемых в нем файлов и подкаталогов. Для отдельных файлов это можно настроить с помощью команды **chattr**.

Ниже приведены примеры установки и проверки флага журналирования для файла **/mnt/gfs2/gfs2_dir/newfile**.

```
[root@roth-01 ~]# chattr +j /mnt/gfs2/gfs2_dir/newfile
[root@roth-01 ~]# lsattr /mnt/gfs2/gfs2_dir
---------j--- /mnt/gfs2/gfs2_dir/newfile
```

Отключение журналирования **/mnt/gfs2/gfs2_dir/newfile**:

```
[root@roth-01 ~]# chattr -j /mnt/gfs2/gfs2_dir/newfile
[root@roth-01 ~]# lsattr /mnt/gfs2/gfs2_dir
------------- /mnt/gfs2/gfs2_dir/newfile
```

Установка флага **j** для каталога автоматически включит журналирование файлов и подкаталогов в его составе. В следующем примере будет установлен флаг **j** для каталога **gfs2_dir** с последующей проверкой его состояния. Затем в этом каталоге будет создан файл **newfile**. Так как флаг уже установлен для каталога, **newfile** его унаследует.

```
[root@roth-01 ~]# chattr -j /mnt/gfs2/gfs2_dir
[root@roth-01 ~]# lsattr /mnt/gfs2
---------j--- /mnt/gfs2/gfs2_dir
[root@roth-01 ~]# touch /mnt/gfs2/gfs2_dir/newfile
[root@roth-01 ~]# lsattr /mnt/gfs2/gfs2_dir
---------j--- /mnt/gfs2/gfs2_dir/newfile
```

4.9. ОБНОВЛЕНИЕ ATIME

Индексному дескриптору каждого файла и каталога соответствуют три метки времени:

- **ctime** — время последнего изменения состояния inode;
- **mtime** — время последнего изменения файла (каталога);
- **atime** — время последнего доступа к файлу (каталогу).

Параметр **atime** обновляется при каждом обращении к файлу.

Так как приложения довольно редко используют информацию **atime**, тратить ресурсы на ее обновление нет смысла. В большинстве случаев можно его полностью отключить или уменьшить частоту обновления.

Частота обновления **atime** контролируется двумя параметрами монтирования:

- **relatime** будет обновлять **atime** при условии, что параметр **atime** в последний раз изменялся раньше чем **mtime** или **ctime**.
- **noatime** полностью отключает обновление метки времени доступа к файлу.

4.9.1. Relatime

Параметр **relatime** откладывает обновление **atime** до тех пор, пока текущее значение **atime** не станет меньше **mtime** или **ctime** (время последнего доступа не может быть меньше времени модификации).

4.9.1.1. Формат команд

```
mount  устройство точка_монтирования -o relatime
```

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог, в который монтируется GFS2.

4.9.1.2. Пример

В этом примере файловая система в **/dev/vg01/lvol0** будет подключена в **/mygfs2**. При этом **atime** будет обновляться только при изменении **mtime** и **ctime**.

```
mount /dev/vg01/lvol0 /mygfs2 -o relatime
```

4.9.2. Noatime

Стандартный параметр монтирования **noatime** отключает обновление **atime**.

4.9.2.1. Формат команд

```
mount устройство точка_монтирования -o noatime
```

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог, в который монтируется GFS2.

4.9.2.2. Пример

В этом примере файловая система в **/dev/vg01/lvol0** будет подключена в **/mygfs2**, и обновление **atime** будет отключено.

```
mount /dev/vg01/lvol0 /mygfs2 -o noatime
```

4.10. ВРЕМЕННАЯ ОСТАНОВКА ФАЙЛОВОЙ СИСТЕМЫ

Запись в файловую систему можно приостановить с помощью команды **dmsetup suspend**. Обычно это делается с целью создания снимка системы. Команда **dmsetup resume** возобновит ее работу.

4.10.1. Формат команд

Приостановить работу

```
dmsetup suspend точка_монтирования
```

Возобновить работу

```
dmsetup resume точка_монтирования
```

точка_монтирования

Каталог подключения файловой системы.

4.10.2. Примеры

Пример приостановки файловой системы **/mygfs2**:

```
# dmsetup suspend /mygfs2
```

Возобновление работы:

```
# dmsetup resume /mygfs2
```

4.11. ПРОВЕРКА ФАЙЛОВОЙ СИСТЕМЫ

Если в файловой системе произошел сбой узлов, их можно будет восстановить с помощью журналов. Но если накопитель был физически отсоединен, или было отключено питание, вероятность повреждения файловой системы не исключена. В таких случаях рекомендуется выполнить проверку **fsck.gfs2**.

ВАЖНО

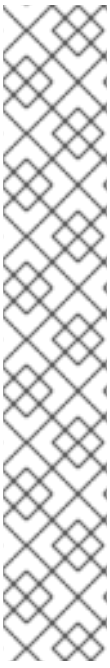
Проверка **fsck.gfs2** должна выполняться в отключенной файловой системе.

ВАЖНО

Не следует выполнять проверку во время загрузки системы, так как **fsck.gfs2** не сможет определить, смонтирована ли файловая система на другом узле кластера. Дождитесь завершения загрузки.

Чтобы предотвратить запуск **fsck.gfs2** во время загрузки, в строке GFS2 в **/etc/fstab** установите нулевые значения в двух последних столбцах.

```
/dev/VG12/lv_svr_home   /svr_home       gfs2
defaults,noatime,nodiratime,noquota     0 0
```



ПРИМЕЧАНИЕ

fsck.gfs2 несколько отличается от **gfs_fsck**:

- **Ctrl**+**C** прервет работу **fsck.gfs2** и покажет запрос, где можно будет выбрать остановить работу команды, пропустить текущий цикл проверки или продолжить работу.
- Параметр **-v** позволяет вывести подробную информацию. Его повторное указание увеличивает детализацию.
- **-q** снижает уровень детализации. Также может быть указан повторно.
- **-n** откроет файловую систему в режиме чтения и будет отвечать **no** на все вопросы. Используется для выявления ошибок без применения изменений **fsck.gfs2**.

Подробную информацию можно найти на справочной странице **fsck.gfs2**.

fsck.gfs2 использует память за пределами операционной системы и ядра. Проверка каждого блока в GFS2 требует примерно 5 бит (5/8 байта). Таким образом, чтобы оценить необходимый размер памяти, надо определить число блоков с помощью **fsck.gfs2** и умножить его на 5/8.

Например, чтобы рассчитать объем памяти, необходимый для выполнения **fsck.gfs2** в файловой системе с блоками размером 4 КБ и общим размером 16 ТБ, надо определить число блоков, разделив размер файловой системы на размер блока:

```
 17592186044416 / 4096 = 4294967296
```

Умножив полученное число на 5/8, получим необходимый размер в байтах:

```
4294967296 * 5/8 = 2684354560
```

Таким образом, для выполнения **fsck.gfs2** потребуется примерно 2.6 ГБ. Если бы размер блока был равен 1 КБ, требовалось бы 11 ГБ.

4.11.1. Формат команд

```
fsck.gfs2 -y устройство
```

-y

yes на все вопросы, то есть **fsck.gfs2** не будет запрашивать ответ перед применением изменений.

устройство

Устройство, где расположена файловая система GFS2.

4.11.2. Пример

Пример восстановления файловой системы в **/dev/testvol/testlv** с автоматическим подтверждением.

```
[root@dash-01 ~]# fsck.gfs2 -y /dev/testvg/testlv
Initializing fsck
Validating Resource Group index.
Level 1 RG check.
(level 1 passed)
Clearing journals (this may take a while)...
Journals cleared.
Starting pass1
Pass1 complete
Starting pass1b
Pass1b complete
Starting pass1c
Pass1c complete
Starting pass2
Pass2 complete
Starting pass3
Pass3 complete
Starting pass4
Pass4 complete
```

```
Starting pass5
Pass5 complete
Writing changes to disk
fsck.gfs2 complete
```

4.12. MOUNT --BIND И КОНТЕКСТНЫЕ ССЫЛКИ

GFS2 не поддерживает контекстные ссылки, использующие переменные для доступа к файлам и каталогам. Для реализации этой функциональности в GFS2 используется опция **bind** команды **mount**.

Опция **bind** позволяет смонтировать часть файловой структуры в другой каталог, не удаляя при этом исходную точку монтирования. Формат команды:

```
mount --bind каталог1 каталог2
```

После этого содержимое первого каталога будет доступно в обоих каталогах. Аналогичным образом можно настроить отдельные файлы.

Пример монтирования **/var/log** в новый каталог **/root/tmp**:

```
[root@menscryfa ~]# cd ~root
[root@menscryfa ~]# mkdir ./tmp
[root@menscryfa ~]# mount --bind /var/log /root/tmp
```

Аналогичного результата можно достичь, добавив запись в файл **/etc/fstab**:

```
/var/log                /root/tmp               none    bind
0 0
```

Чтобы проверить наличие установленной связи, выполните:

```
[root@menscryfa ~]# mount | grep /tmp
/var/log on /root/tmp type none (rw,bind)
```

Рассмотрим еще один пример. Так, в файловых системах, поддерживающих использование контекстных ссылок, **/bin** можно связать с одним из следующих каталогов в зависимости от архитектуры:

```
/usr/i386-bin
/usr/x86_64-bin
/usr/ppc64-bin
```

Аналогичного результата можно достичь с помощью **mount -bind**. Создайте пустой каталог **/bin** и подключите в него вышеперечисленные каталоги. Строка монтирования для первого каталога будет выглядеть так:

```
mount --bind /usr/i386-bin /bin
```

Соответствующая запись в **/etc/fstab**:

```
/usr/1386-bin            /bin               none    bind           0 0
```

mount --bind обеспечивает бóльшую гибкость по сравнению с контекстными ссылками, так как позволяет подключать каталоги в соответствии с пользовательскими критериями (например, используя значение **%fill** для файловой системы). Однако для этого придется написать собственный сценарий.

ПРЕДУПРЕЖДЕНИЕ

Если изначально файловая система была подключена в режиме **rw**, то при создании связи **bind** она тоже будет подключена в режиме **rw** (даже если вы явно укажете **ro**). При этом в каталоге **/proc/mounts** может быть неверно указан режим **ro**.

4.13. MOUNT --BIND И ПОРЯДОК ПОДКЛЮЧЕНИЯ

Файловые системы должны подключаться в определенном порядке. Так, в приведенном ниже примере каталог **/var/log** должен быть смонтирован до того, как будет создана связь между ним и **/tmp**.

```
# mount --bind /var/log /tmp
```

Как определяется порядок:

- Монтирование файловых систем осуществляется в порядке следования в **fstab**. Исключение составляют файловые системы с собственными сценариями **init** и те, которые монтируются с флагом **_netdev**.
- Файловые системы с собственным сценарием **init** монтируются после обработки файла **fstab**.
- Файловые системы, монтируемые с флагом **_netdev**, будут подключены во время инициализации сети.

Если системная конфигурация требует, чтобы монтирование GFS2 осуществлялось в переподключенный каталог, порядок действий в **fstab** должен быть следующим:

1. Сначала монтируются локальные файловые системы.
2. Устанавливается связь с каталогом, в который будет подключена GFS2.
3. Монтирование GFS2 в этот каталог.

Если же наоборот надо перемонтировать локальный каталог в GFS2, порядок в **fstab** не играет никакой роли, так как GFS2 монтируется только на стадии выполнения сценария **init** GFS2. В этом случае **mount --bind** надо добавить в сценарий — тогда связь будет установлена после монтирования GFS2.

Ниже будет приведен пример сценария **init**, который смонтирует два каталога в файловую систему **/mnt/gfs2a**.

Обратите внимание на числа в строке **chkconfig**:

- 345 — уровни выполнения, на которых будет запускаться сценарий;
- 29 — приоритет запуска, то есть сценарий будет запускаться после сценария инициализации GFS2, который имеет приоритет 26;
- 73 — приоритет остановки, то есть этот сценарий будет останавливаться до сценария GFS2, который имеет приоритет 74.

Запуск и остановку можно выполнить вручную с помощью команд **service start** и **service stop**. Например, для запуска сценария **fredwilma** выполните **service fredwilma start**.

Сценарий надо поместить в **/etc/init.d** и присвоить ему такие же разрешения, как и у других сценариев в этом каталоге. Затем выполните **chkconfig on** для его активации на выбранных уровнях выполнения: **chkconfig fredwilma on**.

```
#!/bin/bash
#
# chkconfig: 345 29 73
# description: mount/unmount my custom bind mounts onto a gfs2
subdirectory
#
#
### BEGIN INIT INFO
# Provides:
### END INIT INFO

. /etc/init.d/functions
case "$1" in
  start)
 # In this example, fred and wilma want their home directories
 # bind-mounted over the gfs2 directory /mnt/gfs2a, which has
 # been mounted as /mnt/gfs2a
 mkdir -p /mnt/gfs2a/home/fred &> /dev/null
 mkdir -p /mnt/gfs2a/home/wilma &> /dev/null
 /bin/mount --bind /mnt/gfs2a/home/fred /home/fred
 /bin/mount --bind /mnt/gfs2a/home/wilma /home/wilma
        ;;

  stop)
 /bin/umount /mnt/gfs2a/home/fred
 /bin/umount /mnt/gfs2a/home/wilma
        ;;

  status)
        ;;

  restart)
        $0 stop
        $0 start
        ;;

  reload)
        $0 start
```

```
        ;;
  *)
        echo $"Usage: $0 {start|stop|restart|reload|status}"
        exit 1
esac

exit 0
```

4.14. ФУНКЦИИ ОТЗЫВА GFS2

Целью отзыва является защита целостности файловых систем в кластере. Если модуль ядра GFS2 обнаружил несоответствия в файловой системе после операции ввода-вывода, доступ к ней из кластера будет закрыт. Текущая операция будет остановлена, а последующие операции будут отменены во избежание дальнейших ошибок. Приложения и службы можно остановить вручную, после чего перезагрузить систему и заново смонтировать GFS2. Если это не решило проблему, можно отключить файловую систему на всех узлах кластера и выполнить **fsck.gfs2**. Этот метод является более предпочтительным по сравнению с паникой ядра и изоляцией узла.

Если при загрузке системы запускается сценарий инициализации **gfs2**, а в файле **/etc/fstab** есть записи для GFS2, то файловая система будет заново подключена после перезагрузки. Если же GFS2 была отозвана, то прежде чем заново ее подключить, следует выполнить **fsck.gfs2**. Чтобы предотвратить подключение поврежденной файловой системы во время загрузки, выполните следующее:

1. Отключите сценарий запуска на интересующем узле:

   ```
   # chkconfig gfs2 off
   ```

2. Перезагрузите узел и запустите кластерные службы. GFS2 не будет подключена.

3. Отключите файловую систему на остальных узлах кластера.

4. Выполните **fsck.gfs2** для ее проверки.

5. Заново запустите сценарий на том же узле:

   ```
   # chkconfig gfs2 on
   ```

6. Смонтируйте GFS2 на остальных узлах.

Причиной отзыва GFS2 может послужить неверное число блоков. В частности, это может произойти при удалении файла и соответствующих ему блоков. После этого число блоков будет проверено. Значение, отличное от 1 (должен остаться только дескриптор) сообщает о несоответствии.

Если файловая система была смонтирована с параметром **-o errors=panic**, то ошибки, которые обычно приводят к отзыву, будут вызывать панику и изоляцию узла.

Функция отзыва работает так: ядро отправляет запрос отзыва процессу **gfs_controld**, который вызывает **dmsetup** для настройки проецирования **error**, ограничивающего доступ к файловой системе. Ядру сообщается об успешной изоляции устройства. Именно поэтому GFS2 создается на основе устройств CLVM, так как в противном случае добавление виртуальных устройств Device Mapper было бы невозможно.

Наличие устройства с проецированием **error** гарантирует, что все попытки обращения будут вызывать ошибки, что позволит корректно отключить файловую систему. Поэтому наличие ошибок ввода-вывода в системных журналах после отзыва является нормальным явлением.

Если **dmsetup** не смог создать проекцию, отзыв завершится неудачей. Это может произойти при нехватке памяти на момент инициализации процедуры отзыва.

Иногда причиной отзыва могут стать ошибки ввода-вывода на уровне блочных устройств, а не в GFS2. Поэтому прежде чем принять окончательное решение, стоит проверить журналы.

ГЛАВА 5. ДИАГНОСТИКА КОНФЛИКТОВ GFS2

В этой главе рассматриваются наиболее распространенные проблемы GFS2 и способы их решения.

5.1. НИЗКАЯ ПРОИЗВОДИТЕЛЬНОСТЬ GFS2

Быстродействие GFS2 зависит от разных факторов. В рамках этого документа рассматриваются основные причины снижения производительности и возможные методы ее оптимизации.

5.2. ЗАВИСАНИЕ GFS2 С НЕОБХОДИМОСТЬЮ ПЕРЕЗАГРУЗКИ НА ОДНОМ УЗЛЕ

Если GFS2 зависает, но перезагрузка узла позволяет возобновить работу, причиной может быть ошибка блокировки. Начать следует со сбора статистики.

- Получение информации о блокировках на узле:

  ```
  cat /sys/kernel/debug/gfs2/ФС/glocks >glocks.ФС.узел
  ```

- Получение информации о блокировках DLM:

  ```
  dlm_tool lockdebug -sv имя.
  ```

  Укажите имя пространства блокирования. Это значение можно получить из вывода команды **group_tool**.

- Проверьте вывод **sysrq -t**.

- Проверьте сообщения в **/var/log/messages**.

Завершив сбор информации, можно создать отчет для службы поддержки Red Hat.

5.3. ЗАВИСАНИЕ GFS2 С НЕОБХОДИМОСТЬЮ ПЕРЕЗАГРУЗКИ НА ВСЕХ УЗЛАХ

Если после зависания GFS2 возобновление работы возможно лишь после перезагрузки всех узлов, причина может заключаться в следующем:

- При сбое механизма изоляции файловая система будет заморожена с целью сохранения целостности данных. Проверьте, нет ли в журналах сообщений о сбое механизма изоляции, и проверьте его конфигурацию.

- Проверьте наличие слова **withdraw** в журналах, сообщающего об отзыве файловой системы в результате повреждения или ошибок хранения. Отключите файловую систему, обновите пакет **gfs2-utils** и выполните **fsck**. Создайте отчет для службы поддержки Red Hat, включив содержимое журналов и sosreport.

  Раздел 4.14, «Функции отзыва GFS2» содержит подробную информацию.

- Конфликты могут быть вызваны ошибками блокировки. Создайте отчет для службы поддержки Red Hat, включив собранную статистику (см. Раздел 5.2, «Зависание GFS2 с необходимостью перезагрузки на одном узле»).

5.4. НЕ УДАЕТСЯ ПОДКЛЮЧИТЬ GFS2 НА НОВОМ УЗЛЕ КЛАСТЕРА

Если после добавления нового узла в кластер не удается подключить файловую систему, возможно, число журналов не соответствует числу узлов. Каждому узлу GFS2 должен соответствовать один журнал. Исключение составляют файловые системы, смонтированные с параметром **spectator**, так как в этом случае журнал не требуется. Добавить журналы можно с помощью **gfs2_jadd** (см. Раздел 4.7, «Добавление журналов»).

5.5. ЗАНЯТОЕ ПРОСТРАНСТВО В ПУСТОЙ ФАЙЛОВОЙ СИСТЕМЕ

Если при выполнении **df** в пустой файловой системе обнаружилось, что часть пространства уже занята, скорее всего, оно занято системными журналами. Размер занятого пространства будет равен числу журналов, умноженному на их размер. В небольших файловых системах (меньше 1 ГБ) даже журналы стандартных размеров могут занимать заметную часть пространства.

ГЛАВА 6. GFS2 В КЛАСТЕРЕ PACEMAKER

Далее обсуждаются ключевые этапы конфигурации GFS2 в кластере по управлением Pacemaker.

На всех узлах кластера установите обязательные кластерные программы, пакеты LVM и GFS2. Запустите **cman**, **clvmd** и **pacemaker** на каждом узле, создайте кластер Pacemaker и настройте изоляцию узлов. Подробную информацию о Pacemaker можно найти в документе *Red Hat High Availability и Pacemaker*.

1. Присвойте глобальному параметру **no_quorum_policy** значение **freeze**.

ПРИМЕЧАНИЕ

По умолчанию **no-quorum-policy=stop**, то есть при потере кворума все ресурсы в оставшейся части раздела будут остановлены. Обычно этого должно быть достаточно, но GFS2 отличается тем, что для ее работы необходим кворум. Если кворума нет, файловая система и приложения, обращающиеся к GFS2, не смогут нормально завершить работу. Попытки их остановки завершатся неудачей, что в конце концов приведет к изоляции кластера.

Именно поэтому для GFS2 надо настроить **no-quorum-policy=freeze**. Таким образом, при нарушении кворума остальные ресурсы будут приостановлены до тех пор, пока кворум не будет восстановлен.

```
# pcs property set no-quorum-policy=freeze
```

2. Убедитесь, что в **/etc/lvm/lvm.conf** используется третий тип блокировки, поддерживающий кластерную блокировку. Создайте логический том и отформатируйте его как GFS2. Не забудьте создать достаточное число журналов.

```
# pvcreate /dev/vdb
# vgcreate -Ay -cy cluster_vg /dev/vdb
# lvcreate -L5G -n cluster_lv cluster_vg
# mkfs.gfs2 -j2 -p lock_dlm -t rhel7-demo:gfs2-demo
/dev/cluster_vg/cluster_lv
```

3. Настройте ресурс **clusterfs**.

   Не добавляйте запись в файл **/etc/fstab**, так как файловой системой будет управлять Pacemaker (как кластерным ресурсом). Параметры монтирования можно определить при помощи **options=*параметры*** во время настройки ресурса. Для получения полного списка параметров выполните **pcs resource describe Filesystem**.

   Пример создания кластерного ресурса для файловой системы с параметром **noatime**:

```
# pcs resource create clusterfs Filesystem
device="/dev/cluster_vg/cluster_lv" directory="/var/mountpoint"
fstype="gfs2" "options=noatime" op monitor interval=10s on-
fail=fence clone interleave=true
```

4. Проверьте результат монтирования:

```
# mount |grep /mnt/gfs2-demo
/dev/mapper/cluster_vg-cluster_lv on /mnt/gfs2-demo type gfs2
(rw,noatime,seclabel)
```

5. Дополнительно: перезагрузите узлы, чтобы убедиться, что файловая система монтируется как ожидается.

ПРИЛОЖЕНИЕ A. GFS2_QUOTA

Стандартная функциональность квот в Red Hat Enterprise Linux поддерживается начиная с версии 6.1. Для этого потребуется установить пакет **quota**. В рамках этого документа подразумевается, что управление квотами GFS2 осуществляется с помощью его инструментов (см. Раздел 4.5, «Управление квотами в GFS2»).

В этом приложении обсуждаются основные задачи управления GFS2 при помощи **gfs2_quota**.

A.1. НАСТРОЙКА КВОТ

Для каждого пользователя (UID) и группы (GUID) можно определить два предельных значения: *жесткий лимит* и *гибкий лимит*.

Жесткий лимит определяет абсолютный максимум пространства, доступного пользователю или группе. Нулевое значение снимает ограничения.

Гибкий лимит обычно меньше жесткого . По достижении гибкого лимита файловая система уведомит пользователя. Нулевое значение снимает ограничения.

gfs2_quota позволяет установить лимит. Эту команду нужно выполнить лишь на одном узле после подключения GFS2.

По умолчанию квотирование отключено. Чтобы его включить, при монтировании файловой системы добавьте параметр **quota=on** (см. Раздел A.4, «Включение и отключение квот»).

A.1.1. Формат команд

Настройка квот, жесткий предел

```
gfs2_quota limit -u пользователь -l размер -f точка_монтирования
```

```
gfs2_quota limit -g группа -l размер -f точка_монтирования
```

Настройка квот, гибкий предел

```
gfs2_quota warn -u пользователь -l размер -f точка_монтирования
```

```
gfs2_quota warn -g группа -l размер -f точка_монтирования
```

пользователь

UID или имя пользователя из файла паролей.

группа

GID или название группы.

размер

Новый лимит. В качестве единиц измерения по умолчанию используются мегабайты. Флаги **-k**, **-s** и **-b** позволяют их изменить на килобайты, секторы и блоки файловой системы.

точка_монтирования

Каталог подключения GFS2.

A.1.2. Примеры

В этом примере пользователю *Bert* выделен один гигабайт (1024 МБ) в **/mygfs2**:

```
# gfs2_quota limit -u Bert -l 1024 -f /mygfs2
```

Ниже для группы под номером 21 в **/mygfs2** будет определен гибкий лимит в 50 килобайт.

```
# gfs2_quota warn -g 21 -l 50 -k -f /mygfs2
```

A.2. СТАТИСТИКА ИСПОЛЬЗОВАНИЯ ПРОСТРАНСТВА

С помощью **gfs2_quota get** можно получить информацию о текущих ограничениях и занятом пространстве. Чтобы вывести содержимое файла квот, выполните **gfs2_quota list**.

A.2.1. Формат команд

Просмотр квот пользователя

```
gfs2_quota get -u пользователь -f точка_монтирования
```

Просмотр квот группы

```
gfs2_quota get -g группа -f точка_монтирования
```

Просмотр содержимого файла квот

```
gfs2_quota list -f точка_монтирования
```

пользователь

UID или имя пользователя из файла паролей.

группа

GID или название группы.

точка_монтирования

Каталог подключения GFS2.

A.2.2. Вывод gfs2_quota

Формат вывода **gfs2_quota**:

```
user пользователь: limit:жесткий warn:гибкий value:значение

group группа: limit:жесткий warn:гибкий value:значение
```

По умолчанию в качестве единиц измерения используются мегабайты, что можно изменить с помощью флагов **-k** (килобайты), **-s** (сектора) и **-b** (блоки).

пользователь

Имя пользователя или его идентификатор.

группа

Имя группы или ее идентификатор.

жесткий, гибкий

Предельные значения. Нулевое значение снимает ограничения.

значение

Текущий размер используемого пространства.

A.2.3. Комментарии

Аргумент **-n** команды **gfs2_quota** отключает преобразование UID и GID в имена.

Аргумент **-d** исключает из вывода пространство, занятое скрытыми файлами корневого UID или GID. Обычно используется для сравнения результатов **gfs2_quota** и **du**.

A.2.4. Примеры

Просмотр квот в файловой системе **/mygfs2**.

```
# gfs2_quota list -f /mygfs2
```

Ниже в качестве единиц информации о квотах для группы **users** будут использоваться сектора.

```
# gfs2_quota get -g users -f /mygfs2 -s
```

A.3. СИНХРОНИЗАЦИЯ КВОТ

GFS2 хранит информацию о квотах в специальном файле, который обновляется каждые 60 секунд, а не после каждой записи в файловую систему, что позволяет избежать конфликтов при одновременном изменении файла разными узлами.

По мере приближения к лимиту частота обновления файла будет динамически сокращаться, что уменьшает риск перехода границы. Интервал синхронизации определяется параметром **quota_quantum** и по умолчанию равен 60 секундам (см. Таблица 4.2, «Параметры монтирования GFS2»). Значение **quota_quantum** сбрасывается после отключения файловой системы, но его можно настроить вручную при монтировании при помощи **mount -o remount**.

Для синхронизации файла между автоматическими обновлениями GFS2 используется команда **gfs2_quota sync**.

A.3.1. Формат команд

Синхронизация квот

```
gfs2_quota sync -f точка_монтирования
```

точка_монтирования

Каталог подключения GFS2.

Изменение интервала синхронизации

```
mount -o quota_quantum=интервал,remount устройство точка_монтирования
```

точка_монтирования

Каталог подключения GFS2.

интервал

Интервал синхронизации в секундах. Чем меньше значение, тем лучше точность отслеживания изменений в файле, но тем выше нагрузка.

A.3.2. Примеры

В этом примере выполняется синхронизация квот между узлом и файловой системой **/mygfs2**.

```
# gfs2_quota sync -f /mygfs2
```

Далее интервал обновлений квот для файловой системы **/mnt/mygfs2**, которая монтируется в **/dev/volgroup/logical_volume**, будет увеличен до 1 часа (3600 секунд).

```
# mount -o quota_quantum=3600,remount /dev/volgroup/logical_volume
/mnt/mygfs2
```

A.4. ВКЛЮЧЕНИЕ И ОТКЛЮЧЕНИЕ КВОТ

В GFS2 квоты по умолчанию отключены. Чтобы их включить, при монтировании файловой системы укажите параметр **quota=on**.

A.4.1. Формат команд

```
mount -o quota=on устройство точка_монтирования
```

Чтобы отключить квотирование, укажите **quota=off**. Этот вариант используется по умолчанию.

```
mount -o quota=off устройство точка_монтирования
```

-o quota={on|off}

Включает и отключает квотирование.

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог подключения файловой системы.

A.4.2. Примеры

Монтирование файловой системы на **/dev/vg01/lvol0** в **/mygfs2** с активацией квот:

```
# mount -o quota=on /dev/vg01/lvol0 /mygfs2
```

A.5. СТАТИСТИКА ИСПОЛЬЗОВАНИЯ КВОТ

За использованием дискового пространства можно следить без принуждения жестких ограничений. Для этого следует подключить файловую систему с параметром **quota=account**.

A.5.1. Формат команд

```
mount -o quota=account устройство точка_монтирования
```

-o quota=account

Статистика использования пространства отслеживается на уровне файловой системы.

устройство

Устройство, где расположена файловая система GFS2.

точка_монтирования

Каталог подключения файловой системы.

A.5.2. Пример

В следующем примере файловая система на **/dev/vg01/lvol0** монтируется в каталог **/mygfs2** с поддержкой статистики квот.

```
# mount -o quota=account /dev/vg01/lvol0 /mygfs2
```

ПРИЛОЖЕНИЕ B. ПРЕОБРАЗОВАНИЕ GFS В GFS2

Red Hat Enterprise Linux 6 не поддерживает GFS, поэтому при обновлении Red Hat Enterprise Linux 5 до 6 надо будет преобразовать файловые системы GFS в GFS2 с помощью **gfs2_convert**. Сначала надо будет преобразовать файловые системы и уже после этого обновить операционную систему до Red Hat Enterprise Linux 6.

ПРЕДУПРЕЖДЕНИЕ

Так как процесс преобразования необратим, рекомендуется создать резервную копию файловой системы.

Прежде чем приступить к преобразованию, выполните проверку **gfs_fsck**.

Если преобразование было прервано вследствие потери питания или по другой причине, начните процесс заново. Только после успешного завершения можно еще раз выполнить **fsck.gfs2**.

При преобразовании заполненных файловых систем может оказаться так, что для размещения всех структур данных GFS2 места не хватает. В этом случае место будет освобождено за счет равномерного уменьшения размера журналов.

B.1. ПРЕОБРАЗОВАНИЕ КОНТЕКСТНЫХ ССЫЛОК

GFS2 не поддерживает контекстные ссылки, использующие переменные для доступа к файлам и каталогам. Эту функциональность в GFS2 реализует команда **mount --bind**.

gfs2_convert найдет контекстные ссылки и заменит их одноименными пустыми каталогами. Однако чтобы настроить их точки монтирования, необходимо знать полный путь. Команда **find** поможет его определить.

Так, например, следующая команда покажет пути для переменной **hostname**:

```
[root@smoke-01 gfs]# find /mnt/gfs -lname @hostname
/mnt/gfs/log
```

С помощью **find** также можно найти пути, использующие переменные **mach**, **os**, **sys**, **uid**, **gid**, **jid**. Имя переменной можно указать в формате **@переменная** или **{переменная}**.

Раздел 4.12, «mount --bind и контекстные ссылки» содержит дополнительную информацию.

B.2. ПОРЯДОК ПРЕОБРАЗОВАНИЯ GFS В GFS2

Ниже обсуждается порядок преобразования GFS в GFS2:

1. Создайте резервную копию содержимого GFS.
2. Отключите файловую систему на всех узлах кластера.

3. Выполните проверку **gfs_fsck**.

4. Чтобы начать преобразование, выполните команду **gfs2_convert *ФС***. В процессе работы будут показаны предупреждения и подтверждения.

5. Обновите операционную систему до Red Hat Enterprise Linux 6.

Пример преобразования файловой системы на **/dev/shell_vg/500g**:

```
[root@shell-01 ~]#  /root/cluster/gfs2/convert/gfs2_convert
/dev/shell_vg/500g
gfs2_convert version 2 (built May 10 2010 10:05:40)
Copyright (C) Red Hat, Inc.  2004-2006  All rights reserved.

Examining file system...................
This program will convert a gfs1 filesystem to a gfs2 filesystem.
WARNING: This can't be undone.  It is strongly advised that you:

   1. Back up your entire filesystem first.
   2. Run gfs_fsck first to ensure filesystem integrity.
   3. Make sure the filesystem is NOT mounted from any node.
   4. Make sure you have the latest software versions.
Convert /dev/shell_vg/500g from GFS1 to GFS2? (y/n)y
Converting resource groups...................
Converting inodes.
24208 inodes from 1862 rgs converted.
Fixing file and directory information.
18 cdpn symlinks moved to empty directories.
Converting journals.
Converting journal space to rg space.
Writing journal #1...done.
Writing journal #2...done.
Writing journal #3...done.
Writing journal #4...done.
Building GFS2 file system structures.
Removing obsolete GFS1 file system structures.
Committing changes to disk.
/dev/shell_vg/500g: filesystem converted successfully to gfs2.
```

ПРИЛОЖЕНИЕ C. МОНИТОРИНГ СОБЫТИЙ И ФАЙЛ GLOCKS

Это приложение ориентировано на опытных администраторов, желающих подробнее познакомиться с инструментами отладки GFS2. В частности, здесь будет обсуждаться, как получить информацию о текущих блокировках в GFS2 при помощи средств **debugfs**.

C.1. КАТЕГОРИИ СОБЫТИЙ ТРАССИРОВКИ В GFS2

События трассировки помогают отслеживать поведение файловой системы в реальном времени. В GFS2 различается три категории событий: события блокировки (*glock*), блочные операции (*bmap*) и события журналов. При перехвате события будет вызван соответствующий обработчик. Обычно это используется в целях отладки, особенно если ситуацию, вызвавшую снижение производительности или зависание, можно воспроизвести. Кэширование в GFS2 контролируется механизмом блокировки glocks, поэтому понимание принципов его работы поможет эффективно решать задачи оптимизации производительности файловой системы. Обработчики блочных событий следят за распределением блоков, а обработчики событий журналов, как и следует из названия, следят за тем, как осуществляется запись в журналы.

Благодаря универсальному подходу к проектированию обработчиков, на протяжении жизненного цикла Red Hat Enterprise Linux 6, скорее всего, не потребуется адаптировать API. С другой стороны, важно помнить, что это все-таки отдельный отладочный интерфейс, не входящий в комплект API Red Hat Enterprise Linux 6, поэтому Red Hat не может гарантировать, что обработчики не будут модифицироваться.

Совместное использование обработчиков GFS2 с другими средствами диагностики (наподобие **blktrace**) поможет получить объективное представление о производительности системы. Несмотря на свою облегченную структуру, обработчики генерируют большие объемы информации за короткий период времени. Чтобы сократить нагрузку, рекомендуется использовать фильтры и отслеживать лишь необходимые события.

C.2. ОБРАБОТЧИКИ

Если файловая система **debugfs** смонтирована в **/sys/kernel/debug**, обработчики можно будет найти в **/sys/kernel/debug/tracing/**. Подкаталог **events** содержит полный список событий, а если был загружен модуль **gfs2** — то и каталог **gfs2** с отдельными подкаталогами для каждого события. Содержимое **/sys/kernel/debug/tracing/events/gfs2** может выглядеть примерно так:

```
[root@chywoon gfs2]# ls
enable            gfs2_bmap       gfs2_glock_queue         gfs2_log_flush
filter            gfs2_demote_rq  gfs2_glock_state_change  gfs2_pin
gfs2_block_alloc  gfs2_glock_put  gfs2_log_blocks          gfs2_promote
```

Чтобы включить все обработчики событий GFS2, выполните:

```
[root@chywoon gfs2]# echo -n 1
>/sys/kernel/debug/tracing/events/gfs2/enable
```

Чтобы включить конкретный обработчик, перейдите в его подкаталог и измените **enable** так же, как это сделано выше. Аналогично можно настроить **filter** для всех обработчиков в **/sys/kernel/debug/tracing/events/gfs2/filter** или по отдельности в подкаталогах.

Обработчики возвращают данные в двоичном формате или ASCII. Двоичный интерфейс здесь не рассматривается, а для просмотра данных ASCII есть два способа: первый заключается в выводе содержимого кольцевого буфера, а второй возвращает все данные. Для просмотра данных в буфере выполните:

```
[root@chywoon gfs2]# cat /sys/kernel/debug/tracing/trace
```

Если интересующий процесс уже выполняется на протяжении какого-то времени, команда вернет последние данные в буфере. Второй способ позволяет следить за событиями в реальном времени, выполняя чтение из файла **/sys/kernel/debug/tracing/trace_pipe**. По мере чтения событий они будут удаляться из файла. Формат вывода в обоих случаях идентичен.

Наблюдение за процессами можно вести при помощи утилиты **trace-cmd** (см. Раздел C.10, «Дополнительные ресурсы»), которая перехватывает события аналогично тому, как это делает **strace**, и поможет понять, что делает процесс в заданное время.

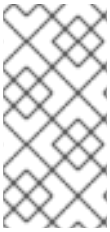
C.3. GLOCKS

GFS2 принципиально отличается от других файловых систем тем, что в ней реализован механизм блокировки glocks. На уровне кода glock представляет собой структуру данных, объединяющую блокировку DLM с функциями кэширования в конечный автомат с одним состоянием. Каждый glock связан с одной блокировкой DLM, состояние которой хранится в кэше, чтобы в случае повторного обращения с того же узла не требовалось заново вызывать DLM. Блокировки glock подразделяются на две основных категории в зависимости от того, кэшируют ли они метаданные или нет. Так, блокировки индексных дескрипторов и групп ресурсов кэшируют метаданные. Более того, при блокировке дескрипторов кэшируются и обычные данные, что обуславливает довольно сложную логику этой блокировки.

Таблица C.1. Режимы блокировки glock и DLM

Режим glock	Режим DLM	Описание
UN	IV/NL	Не заблокировано (блокировка DLM не назначена или назначена в режиме NL, о чем свидетельствует установленный флаг «I»).
SH	PR	Совместный доступ (защищенное чтение).
EX	EX	Монопольная блокировка.
DF	CW	Параллельная запись. Используется для прямого ввода-вывода.

Glock остается в памяти до тех пор, пока он не будет освобожден (по запросу другого узла или виртуальной машины). После освобождения он будет удален из хэш-таблицы glock. В свою очередь, при создании glock изначально он не связан с блокировкой DLM, но как только будет сделан первый вызов к DLM, она будет назначена, о чем свидетельствует установленный флаг «I» (см. Таблица C.4, «Флаги glock»). Пока glock удерживается, блокировка DLM тоже будет удерживаться как минимум в состоянии NL (Null). Понижение статуса NL снимает блокировку DLM и выполняется только при освобождении glock.

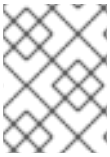


ПРИМЕЧАНИЕ

В Red Hat Enterprise Linux 5 снятие блокировки DLM, связанной с активным glock, не запрещалось. В Red Hat Enterprise Linux 6 это поведение изменилось вследствие интеграции модуля **`lock_dlm`** в GFS2, поэтому блокировка DLM освобождается одновременно с glock.

Один glock может обрабатывать несколько запросов с вышестоящих уровней файловой системы. В целях защиты критических секций кода, добавление запросов в очередь осуществляется через системные вызовы GFS2.

В основе реализации конечного автомата glock лежит рабочая очередь — workqueue. В интересах производительности лучше всего подошли бы микропотоки tasklet, однако в текущей реализации ввод-вывод обрабатывается в контексте, не допускающем их использование.



ПРИМЕЧАНИЕ

Механизм workqueue использует свои события трассировки, которые можно комбинировать с обработчиками GFS2.

Таблица C.2, «Кэширование glock» показывает, в каких режимах glock разрешается кэширование и допускается использование несогласованного кэша. Это касается блокировок inode и групп ресурсов (хотя при блокировке групп ресурсов кэшируются только метаданные).

Таблица C.2. Кэширование glock

Режим glock	Данные	Метаданные	Несогласованные данные	Несогласованные метаданные
UN	нет	нет	нет	нет
SH	да	да	нет	нет
DF	нет	да	нет	нет
EX	да	да	да	да

C.4. ПРОСМОТР СТАТУСА GLOCK СРЕДСТВАМИ DEBUGFS

`Debugfs` предоставляет интерфейс для визуализации внутреннего состояния блокировок. Секция блокировки начинается с «G:». Следующие за ней строки начинаются с пробела и содержат подробную информацию (например, «H:» обозначает запрос, удерживающий glock, «I:» — inode, «R:» — группу ресурсов).

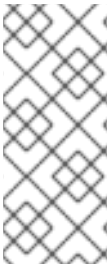
```
G:  s:SH n:5/75320 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:EX n:3/258028 f:yI t:EX d:EX/0 a:3 r:4
 H: s:EX f:tH e:0 p:4466 [postmark] gfs2_inplace_reserve_i+0x177/0x780
[gfs2]
 R: n:258028 f:05 b:22256/22256 i:16800
G:  s:EX n:2/219916 f:yfI t:EX d:EX/0 a:0 r:3
 I: n:75661/219916 t:8 f:0x10 d:0x00000000 s:7522/7522
```

```
G:  s:SH n:5/127205 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:EX n:2/50382 f:yfI t:EX d:EX/0 a:0 r:2
G:  s:SH n:5/302519 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/313874 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/271916 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
G:  s:SH n:5/312732 f:I t:SH d:EX/0 a:0 r:3
 H: s:SH f:EH e:0 p:4466 [postmark] gfs2_inode_lookup+0x14e/0x260 [gfs2]
```

Это фрагмент файла, который будет создан при выполнении **`cat /sys/kernel/debug/gfs2/unity:myfs/glocks >my.lock`** во время теста postmark на узле GFS2. Здесь показаны блокировки, демонстрирующие наиболее интересные свойства glock.

Возможные состояния glock включают EX (exclusive), DF (deferred), SH (shared) и UN (unlocked), каждое из которых соответствует определенному режиму блокировки DLM. Исключение составляет glock в состоянии UN, так как он может быть вообще не связан с блокировкой DLM или связан в режиме NL (о чем будет свидетельствовать установленный флаг «l» в поле «f:»).

Обратите внимание на поле «n:*тип*/*номер*». Так, в первой строке в примере можно увидеть «n:5/75320», где 5 означает тип **`iopen`**, а 75320 — номер файлового дескриптора. Для типов 2 и 5 за наклонной чертой следует номер дескриптора на диске.



ПРИМЕЧАНИЕ

Число в поле «n:» представлено в шестнадцатеричном формате, в то время как в выводе обработчиков событий трассировки эти же числа будут показаны в десятичном формате. Это сделано намеренно для того, чтобы их можно было сравнить с выводом других инструментов анализа производительности, таких как **`blktrace`** и **`stat`**(1).

Таблица C.4, «Флаги glock» и Таблица C.5, «Флаги удерживания glock» содержат описание флагов glock и запросов блокировки. Содержимое блоков состояния блокировки в настоящий момент не может быть проверено средствами **`debugfs`**.

Ниже приведено описание типов glock.

Таблица C.3. Типы glock

Номер	Тип	Описание
1	trans	Блокировка транзакции
2	inode	Индексный дескриптор
3	rgrp	Метаданные группы ресурсов
4	meta	Суперблок
5	iopen	Идентифицирует открытый дескриптор

Номер	Тип	Описание
6	flock	Системный вызов **flock**(2)
8	quota	Операции с квотами
9	journal	Мьютекс журнала

Ниже приведена таблица флагов glock. Отдельно стоит отметить флаг «l» (locked), который ограничивает доступ к glock во время изменения его состояния. Этот флаг устанавливается непосредственно перед тем, как будет запрошена блокировка DLM, и снимается после ее установки.

Таблица C.4, «Флаги glock» содержит список возможных значений в поле «f:» в строке «G:».

Таблица C.4. Флаги glock

Флаг	Имя	Описание
d	Pending demote	Получен запрос снижения режима блокировки, но glock уже удерживается, и время минимального обслуживания не истекло.
D	Demote	Есть запрос снижения режима блокирования (локальный или удаленный).
f	Log flush	Прежде чем освободить glock, необходимо сохранить журнал.
F	Frozen	Идет восстановление — ответы удаленных узлов игнорируются.
i	Invalidate in progress	Сброс страниц кэша под блокировкой.
I	Initial	Выбранному glock назначена блокировка DLM.
l	Locked	Glock в процессе изменения состояния.
L	LRU	Glock в списке LRU.
o	Object	Означает, что glock связан с объектом. Объекты могут быть разных типов. Так, тип 2 означает дескриптор, 3 — группу ресурсов.
p	Demote in progress	Glock в процессе обработки запроса снижения режима блокировки.

Флаг	Имя	Описание
q	Queued	Устанавливается при наличии ожидающих запросов и снимается, если запросов нет. Используется алгоритмом расчета минимального времени удерживания glock.
r	Reply pending	Ответ, полученный от удаленного узла, ожидает обработки.
y	Dirty	Прежде чем освободить glock, необходимо сохранить данные.

Чтобы уменьшить риск конкуренции из-за частого перехвата glock разными запросами, для каждого glock устанавливается минимальное время, на протяжении которого он может удерживаться одним запросом. При получении удаленного запроса, несовместимого с текущим режимом блокировки, будет установлен флаг D (demote) или d (demote pending) в зависимости от того, когда истекает минимальное время обслуживания.

Если интервал подошел к концу, будет установлен флаг D, и запрашиваемый режим блокировки будет записан. Как только в очереди не останется активных блокировок, статус glock будет понижен. Если же минимальное время обслуживания не истекло, будет установлен флаг ожидания d, который будет изменен на D, когда оно подойдет к концу.

Флаг «I» (Initial) будет установлен, как только с glock будет связана блокировка DLM, и сброшен только после снятия блокировки DLM и освобождения glock.

C.5. ЗАПРОСЫ БЛОКИРОВКИ

Таблица C.5, «Флаги удерживания glock» содержит список флагов в поле «f:» строке «H:».

Таблица C.5. Флаги удерживания glock

Флаг	Имя	Описание
a	Async	Не ждать результата glock (будет запрошен позднее).
A	Any	Принимает любой совместимый режим блокировки.
c	No cache	Если не заблокировано, сразу снизить режим блокировки DLM.
e	No expire	Игнорирует последующие запросы снятия блокировки.
E	Exact	Требуется конкретный режим блокировки.
F	First	Первый запрос, захвативший блокировку.
H	Holder	Блокировка установлена.
p	Priority	Ставит запрос блокировки в начало очереди.

Флаг	Имя	Описание
t	Try	Пробная блокировка.
T	Try 1CB	Пробная блокировка с ответом.
W	Wait	Ожидание обработки запроса.

Особое внимание следует обратить на два флага: «W» отмечает ожидающий запрос, а «H» — активную блокировку. В начале списка идут запросы, удерживающие блокировку, за ними следуют ожидающие запросы.

Если удерживающих запросов нет, то первый запрос в списке инициирует изменение состояния glock. Но так как запросы снижения статуса glock имеют более высокий приоритет по сравнению с вызовами из файловой системы, это может повлиять на порядок обработки.

Пробная блокировка (флаг «t») проверяет состояние glock и пропускает его, если он занят. Флаг «T» (**`try 1CB`**) отличается лишь тем, что DLM отправит свой запрос удерживаемому glock. В частности, **`try 1CB`** используется вместе с **`iopen`** и помогает выбрать узел, который освободит inode. По умолчанию **`iopen`** удерживается в разделяемом режиме, но как только **`i_nlink`** достигнет нулевого значения, и будет вызван **`->delete_inode`**(), будет установлен флаг «T» и запрошен монопольный режим. Если удалось установить монопольный режим, то inode будет освобожден. Если же блокировка удерживается другим узлом, для его glock будет установлен флаг ожидания «D», и он снова будет проверен при вызове **`->drop_inode`**().

Как только **`i_nlink`** дескриптора уменьшится до нуля, в битовой карте групп ресурсов он будет отмечен как ожидающий освобождения, но все еще используемый. При следующем чтении битовой карты это будет обнаружено, после чего будет предпринята очередная попытка его освобождения. В результате дескриптор будет освобожден тем узлом, на котором будет сделан последний вызов **`close`**(), так как ничто не будет препятствовать завершению операции.

C.6. ОБРАБОТЧИКИ СОБЫТИЙ GLOCK

Обработчики событий glock помогают провести анализ вывода blktrace, отследить состояние кэша, убедиться, что запросы ввода-вывода обрабатываются в подходящем режиме блокировки, и проверить уровень конкуренции блокировок.

`gfs2_glock_state_change` является одним из ключевых обработчиков: он следит за изменением состояния glock с момента создания и до его освобождения. В свою очередь, освобождение glock, которое знаменуется переходом из режима NL в незаблокированное состояние, контролируется обработчиком **`gfs2_glock_put`**. Как уже говорилось, на время изменения состояния устанавливается флаг «l» (locked), который предотвращает попытки перехвата блокировки. Остальные запросы будут ожидать в состоянии «W». После успешного изменения состояния блокировку можно будет перехватить.

`gfs2_demote_rq` отслеживает запросы снижения режима блокировки (локальные и удаленные). Если на узле достаточно памяти, то локальные запросы будут появляться редко и чаще всего будут иметь отношение к umount и операциям освобождения памяти, в то время как количество удаленных запросов напрямую отражает уровень конкуренции за доступ к inode или группе ресурсов.

Если блокировка предоставлена успешно, генерируется событие **`gfs2_promote`**. Обычно это происходит на последних стадиях изменения состояния glock или по запросу блокировки,

которую можно установить сразу в силу того, что glock уже кэширует ее в подходящем режиме. Первый запрос, перехвативший glock, будет отмечен флагом «F» (First).

C.7. ОТСЛЕЖИВАНИЕ БЛОЧНЫХ ОПЕРАЦИЙ

В основе работы любой файловой системы лежит манипулирование блоками. Для хранения информации об использовании пространства GFS2 использует традиционную битовую карту, где состояние каждого блока представлено двумя битами. При помощи обработчиков событий файловой системы можно определить, сколько времени занимают операции распределения и сопоставления блоков.

gfs2_bmap вызывается дважды для одной операции с картой блоков: первый раз для просмотра запроса, второй раз — для отображения результата. Таким образом, можно легко определить, сколько времени уходит на отведение блоков в разных сегментах файловой системы или даже в разных файлах. Этот обработчик также помогает сравнить средний размер полученных экстентов с запрашиваемым.

gfs2_block_alloc следит не только за выделением блоков, но и за их освобождением. Каждый занятый блок связан с индексным дескриптором, что позволяет определить, каким файлам отведены те или иные блоки. В комбинации с результатами теста **blktrace**, который анализирует производительность ввода-вывода, **gfs2_block_alloc** поможет определить дескрипторы, и как следствие — файлы, которым принадлежат проблемные блоки.

C.8. ОБРАБОТЧИКИ СОБЫТИЙ ЖУРНАЛОВ

gfs2_pin следит за добавлением блоков в журнал и их удалением, а **gfs2_log_flush** — за временем, потраченным на запись транзакций в журнал. Эти обработчики обычно используются при анализе производительности журналирования.

gfs2_log_blocks контролирует зарезервированные блоки в журнале, что, в частности, поможет определить, соответствует ли размер журнала текущему объему нагрузки.

gfs2_ail_flush (Red Hat Enterprise Linux 6.2 и выше) отслеживает время записи списка AIL (Active Item List), содержащего изменения журнала, которые еще не были сохранены. Список AIL освобождается периодически с целью освобождения места или по запросу fsync.

C.9. СТАТИСТИКА GLOCK

GFS2 предоставляет инструменты для сбора статистики, анализ которой поможет идентифицировать причины снижения производительности и оптимизировать работу файловой системы.

GFS2 использует два счетчика:

- **dcount** отслеживает число запросов к DLM и помогает оценить, на основе какого количества исходных данных рассчитывалось среднее значение и дисперсия цикла блокировки DLM;
- **qcount** подсчитывает число вызовов **syscall**, тем самым отслеживая число запросов блокировки в очереди glock. Желательно, чтобы это число превышало значение **dcount**.

Также статистику производительности GFS2 можно извлечь из трех пар значений (среднего и дисперсии), которые рассчитываются по сглаженному экспоненциальному алгоритму аналогично тому, как это делается при расчете времени полного цикла в сетевых протоколах. Их значения

приведены в наносекундах.

- srtt/srttvar: продолжительность полного цикла неблокирующих запросов DLM;
- srttb/srttvarb: продолжительность полного цикла блокирующих запросов DLM;
- irtt/irttvar: интервал между запросами DLM.

Неблокирующий запрос сразу возвращает результат независимо от состояния блокировки DLM. Это происходит в нескольких случаях: если запрашиваемая блокировка свободна или установлена в режиме NL (то есть ее можно перехватить), установлена в монопольном режиме, или если установлен флаг пробного запроса блокировки («t», «T»). Все остальные запросы являются блокирующими.

Чем реже делаются запросы к DLM (в силу высокого попадания в кэш), и чем быстрее обслуживаются оба типа запросов, тем выше эффективность работы файловой системы.

Статистика GFS2 хранится в файлах **`sysfs`**:

- **`glstats`** — его формат напоминает файл **`glocks`**, но на один glock выделена одна строка, в которой приведена статистика по рассмотренным выше критериям.
- **`lkstats`** содержит статистику glock по процессорным ядрам. Каждая строка содержит одну из восьми перечисленных характеристик glock, то есть для одного glock требуется 8 строк (три пары средних значений и дисперсии плюс два счетчика). В столбцах перечислены процессорные ядра.

C.10. ДОПОЛНИТЕЛЬНЫЕ РЕСУРСЫ

За дальнейшей информацией о **`glocks`** и мониторинге событий GFS2 обратитесь к следующим ресурсам:

- Материалы в этом приложении были частично заимствованы из доклада Стива Уайтхауза на симпозиуме Linux 2009: http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=blob;f=Documentation/filesystems/gfs2-glocks.txt;h=0494f78d87e40c225eb1dc1a1489acd891210761;hb=HEAD.
- Описание внутренних правил блокирования: http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=blob;f=Documentation/filesystems/gfs2-glocks.txt;h=0494f78d87e40c225eb1dc1a1489acd891210761;hb=HEAD.
- Отслеживание событий: http://git.kernel.org/?p=linux/kernel/git/torvalds/linux-2.6.git;a=blob;f=Documentation/trace/events.txt;h=09bd8e9029892e4e1d48078de4d076e24eff3dd2
- Описание **`trace-cmd`**: http://lwn.net/Articles/341902/.

ПРИЛОЖЕНИЕ D. ИСТОРИЯ ПЕРЕИЗДАНИЯ

Издание 7.1-3.2 **Fri Mar 6 2015** **Yuliya Poyarkova**
Перевод на русский язык.

Издание 7.1-3.1 **Fri Mar 6 2015** **Yuliya Poyarkova**
Синхронизация XML 7.1-3

Издание 7.1-3 **Tue Dec 16 2014** **Steven Levine**
Добавлена функция сортировки на странице приветствия RHEL 6.

Издание 7.0-9 **Wed Oct 8 2014** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.6.

Издание 7.0-8 **Thu Aug 7 2014** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.6 Beta.

Издание 7.0-4 **Thu Jul 17 2014** **Steven Levine**
BZ #1102591
Добавлено описание конфигурации GFS2 в кластере Pacemaker.

Издание 7.0-3 **Wed Jul 16 2014** **Steven Levine**
BZ #1035119
Обновлена таблица с флагами glock, добавлена информация о статистике glock.

Издание 7.0-1 **Thu Jun 5 2014** **Steven Levine**
Черновой вариант для версии 6.6.

Издание 6.0-6 **Wed Nov 13 2013** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.5.

Издание 6.0-5 **Fri Sep 27 2013** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.5 Beta.

Издание 6.0-3 **Fri Sep 27 2013** **Steven Levine**
BZ #960841
Уточнение о неэффективности использования SELinux.

Издание 6.0-1 **Fri Sep 06 2013** **Steven Levine**
Примечание о Samba и GFS2.

Издание 5.0-7 **Mon Feb 18 2013** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.4.

Издание 5.0-5 **Mon Nov 26 2012** **Steven Levine**
Публикация для Red Hat Enterprise Linux 6.4 Beta.

Издание 5.0-4 **Tue Nov 13 2012** **Steven Levine**
BZ #860324
Обновления и уточнения в главе о конфигурации GFS2.

BZ #807057
Рекомендация связи с представителем Red Hat по вопросу соответствия конфигурации требованиям.

Издание 5.0-1 **Mon Oct 15 2012** **Steven Levine**
Обновлена глава о функциональных особенностях.

Издание 4.0-2 **Thu Mar 28 2012** **Steven Levine**

Версия для Red Hat Enterprise Linux 6.3.

Издание 4.0-1 **Thu Mar 28 2012** **Steven Levine**

BZ #782482, BZ #663944

Добавлена глава о конфигурации GFS2.

BZ#757742

Уточнение о необходимости использования CLVM с GFS2.

BZ #786621

Исправления опечаток.

Издание 3.0-2 **Thu Dec 1 2011** **Steven Levine**

Подготовлено к выпуску Red Hat Enterprise Linux 6.2

Издание 3.0-1 **Mon Sep 19 2011** **Steven Levine**

Исходная версия для Red Hat Enterprise Linux 6.2 Beta.

BZ #704179

Информация о поддержке **tunegfs2**.

BZ #712390

Добавлено приложение об обработчиках событий GFS2.

BZ #705961

Исправления опечаток.

Издание 2.0-1 **Thu May 19 2011** **Steven Levine**

Исходная версия документа для Red Hat Enterprise Linux 6.1

BZ #549838

Добавлено описание стандартных квот в Red Hat Enterprise Linux 6.1.

B Z#608750

Откорректировано описание функций отзыва.

BZ #660364

Исправлен максимальный размер GFS2.

BZ #687874

Добавлена глава о диагностике GFS2.

BZ #664848

Дополнения относительно преобразования GFS в GFS2.

Издание 1.0-1 **Wed Nov 15 2010** **Steven Levine**

Исходная версия для Red Hat Enterprise Linux 6

ПРЕДМЕТНЫЙ УКАЗАТЕЛЬ

Символы

Особенности конфигурации, Функциональные особенности и конфигурация GFS2
блокировка Posix, Блокировка Posix
блокировка узлов, Блокировка узлов GFS2
введение, Введение
    целевая аудитория, Целевая аудитория

ведение журналов, Журналирование данных
возможности, новые и измененные, Новые и измененные функции
дисковые квоты
    введение квот для групп, Установка квот для групп пользователей
    гибкий лимит, Установка квот для пользователей
    добавление, Настройка дисковых квот
    дополнительные ресурсы, Дополнительные ресурсы
    жесткий лимит, Установка квот для пользователей
    квоты пользователей, Установка квот для пользователей
    управление, Repquota
        команда quotacheck, проверка, Quotacheck
        статистика, Repquota

добавление журналов для файловой системы, Добавление журналов
зависание файловой системы при отключении, Особенности подключения GFS2
исходные задачи
    настройка, исходная, Первоначальная конфигурация

квоты
    добавление
        quotacheck, выполнение, Создание базы данных квот
        создание файлов квот, Создание базы данных квот

команда fsck.gfs2, Проверка файловой системы
команда gfs2_grow, Увеличение размера файловой системы
команда gfs2_jadd, Добавление журналов
команда gfs2_quota, gfs2_quota
команда mkfs, Создание файловой системы
команда mount, Монтирование файловой системы
команда quotacheck
    проверка точности квот, Quotacheck

команда umount, Отключение файловой системы

контекстные ссылки, mount --bind и контекстные ссылки
  преобразование GFS в GFS2, Преобразование контекстных ссылок

конфигурация, исходная, Начало работы
  предварительные требования, Предварительные требования

конфигурация, подготовка, Проектирование структуры GFS2
максимальный размер GFS2, Обзор GFS2
максимальный размер, GFS2, Обзор GFS2
монтирование связей, mount --bind и контекстные ссылки
монтирование файловой системы, Монтирование файловой системы, Особенности подключения GFS2
настройка, исходная
  исходные задачи, Первоначальная конфигурация

обзор, Обзор GFS2
  возможности, новые и измененные, Новые и измененные функции
  предварительная подготовка, Проектирование структуры GFS2

обработчики, Мониторинг событий и файл glocks
оптимизация производительности, Оптимизация производительности
оптимизация, производительность, Оптимизация производительности
остановка записи в файловую систему, Временная остановка файловой системы
отзыв, GFS2, Функции отзыва GFS2
отзывы
  контактная информация, Отзывы и предложения

отключение файловой системы, Отключение файловой системы, Особенности подключения GFS2
отключение, зависание файловой системы, Особенности подключения GFS2
параметр acl, Монтирование файловой системы
параметр quota_quantum, Quotasync, Синхронизация квот
параметры gfs2_jadd, Полная форма
предварительные требования
  конфигурация, исходная, Предварительные требования

предисловие (см. введение)
проверка файловой системы, Проверка файловой системы
создание файловой системы, Создание файловой системы
таблица параметров mkfs.gfs2, Полный список параметров
таблица параметров mount, Полная форма
таблица параметров увеличения размера файловой системы, Полная форма
таблицы
  параметры gfs2_jadd, Полная форма
  параметры mkfs.gfs2, Полный список параметров

параметры монтирования, Полная форма
параметры увеличения размера файловой системы, Полная форма

типы glock, Статистика блокировок, Просмотр статуса glock средствами debugfs
увеличение размера файловой системы, Увеличение размера файловой системы
управление GFS2, Управление GFS2
управление квотами, Управление квотами в GFS2, Настройка квот, gfs2_quota
настройка квот, Настройка квот
принудительный режим, Включение и отключение квот
просмотр квот, Статистика использования пространства
синхронизация квот, Quotasync, Синхронизация квот
статистика квот, Статистика использования квот

файл debugfs, Статистика блокировок
файловая система
atime, настройка обновлений, Обновление atime
монтирование с параметром noatime , Noatime
монтирование с параметром relatime , Relatime

bind, mount --bind и контекстные ссылки
ведение журналов, Журналирование данных
добавление журналов, Добавление журналов
контекстные ссылки, mount --bind и контекстные ссылки
монтирование, Монтирование файловой системы, Особенности подключения GFS2
остановка, Временная остановка файловой системы
отключение, Отключение файловой системы, Особенности подключения GFS2
порядок подключения, mount --bind и порядок подключения
проверка, Проверка файловой системы
создание, Создание файловой системы
увеличение, Увеличение размера файловой системы
управление квотами, Управление квотами в GFS2, Настройка квот, gfs2_quota
настройка квот, Настройка квот
принудительный режим, Включение и отключение квот
просмотр квот, Статистика использования пространства
синхронизация квот, Quotasync, Синхронизация квот
статистика квот, Статистика использования квот

флаги glock, Статистика блокировок, Просмотр статуса glock средствами debugfs
флаги удерживания glock, Статистика блокировок, Запросы блокировки
целевая аудитория, Целевая аудитория

A

atime, настройка обновлений, Обновление atime

монтирование с параметром noatime , Noatime
монтирование с параметром relatime , Relatime

D

debugfs, Мониторинг событий и файл glocks

G

GFS2
atime, настройка обновлений, Обновление atime
монтирование с параметром noatime , Noatime
монтирование с параметром relatime , Relatime
отзыв, Функции отзыва GFS2
работа, Функциональные особенности и конфигурация GFS2
управление, Управление GFS2
управление квотами, Управление квотами в GFS2, Настройка квот, gfs2_quota
настройка квот, Настройка квот
принудительный режим, Включение и отключение квот
просмотр квот, Статистика использования пространства
синхронизация квот, Quotasync, Синхронизация квот
статистика квот, Статистика использования квот
функциональные особенности, Функциональные особенности и конфигурация GFS2

glock, Мониторинг событий и файл glocks

M

mount --bind
порядок подключения, mount --bind и порядок подключения

Q

quota=режим, Настройка квот
quotacheck , Создание базы данных квот