

raidctl(8)

НАЗВАНИЕ

raidctl - утилита конфигурирования для драйвера диска RAIDframe

СИНТАКСИС

```
raidctl [-v] -a компонент устройство
raidctl [-v] -A [yes | no | root] устройство
raidctl [-v] -B устройство
raidctl [-v] -c файл_конфигурации
raidctl [-v] -C файл_конфигурации
raidctl [-v] -f компонент устройство
raidctl [-v] -F компонент устройство
raidctl [-v] -g компонент устройство
raidctl [-v] -i устройство
raidctl [-v] -I серийный_номер устройство
raidctl [-v] -p устройство
raidctl [-v] -P устройство
raidctl [-v] -r компонент устройство
raidctl [-v] -R компонент устройство
raidctl [-v] -s устройство
raidctl [-v] -S устройство
raidctl [-v] -u устройство
```

ОПИСАНИЕ

raidctl - это пользовательская программа для управления устройствами [raid\(4\)](#), дисковыми устройствами RAIDframe. **raidctl**, в основном, используется для динамического конфигурирования и деконфигурирования дисков RAIDframe. Дополнительную информацию об устройствах RAIDframe см. на странице справочного руководства [raid\(4\)](#).

Предполагается, что читатель хотя бы поверхностно знаком с RAID и соответствующей терминологией.

Опции команды **raidctl** следующие:

-a компонент устройство

Добавить компонент как диск горячей замены для устройства.

-A yes устройство

Сделать RAID-набор автоматически конфигурируемым. RAID-набор будет автоматически конфигурироваться при загрузке перед монтированием корневой файловой системы. Учтите, что все компоненты набора должны быть обозначены в метке диска как RAID.

-A no устройство

Отключить автоматическое конфигурирование для RAID-набора.

-A root устройство

Сделать RAID-набор автоматически конфигурируемым, а также пометить его как пригодный для корневого раздела. Сконфигурированный таким образом RAID-набор будет использоваться вместо загрузочного диска в качестве корневого устройства. Все компоненты набора должны быть обозначены в метке диска как RAID. Учтите, что загружаемое ядро пока что не может находиться на RAID-наборе.

-B устройство

Инициировать обратное копирование восстановленных данных с резервного диска на исходный. Это делается после сбоя компонента и воссоздания сбойного диска на резервном.

-с файл_конфигурации

Сконфигурировать устройство RAIDframe в соответствии с конфигурацией в указанном файле.

Описание содержимого **файла_конфигурации** представлено далее.

-С файл_конфигурации

Аналогично **-с**, но выполняет конфигурирование принудительно. Это необходимо при первоначальном конфигурировании RAID-набора.

-f компонент устройство

Пометить указанный **компонент** как сбойный, но не инициировать пересоздание этого компонента.

-F компонент устройство

Пометить как сбойный указанный **компонент** устройства и немедленно начать пересоздание сбояного диска на имеющемся диске горячего резерва. Это - один из механизмов, используемый для запуска процесса пересоздания, если на компоненте действительно произошел аппаратный сбой.

-g компонент устройство

Получить метку указанного **компонента**.

-i устройство

Инициализировать устройство RAID. В частности, (переписать) информацию о четности на выбранном устройстве. Это ОБЯЗАТЕЛЬНО делать для всех RAID-наборов перед разметкой RAID-устройства и созданием на нем файловых систем.

-I серийный_номер устройства

Инициализировать метки на всех компонентах устройства. **Серийный_номер** используется как один из ключей при определении того, принадлежит ли определенный набор компонентов к одному RAID-набору. Хотя это и не проверяется, в разных RAID-наборах надо использовать разные серийные номера. Этот шаг **НЕОБХОДИМО** выполнять при создании нового RAID-набора.

-p устройство

Проверить статус четности на RAID-наборе. Команда выдает сообщение о состоянии и успешно завершает работу, если информация о четности актуальна.

-P устройство

Проверить статус четности на RAID-наборе и проинициализировать (переписать) информацию о четности, если она не является заведомо актуальной. Эта опция обычно используется после сбоя системы (и перед вызовом **fsck(8)**), чтобы гарантировать целостность информации о четности.

-r компонент устройство

Удалить резервный диск, заданный как **компонент**, из набора доступных резервных компонентов.

-R компонент устройство

Пометить указанный **компонент** как сбойный, при необходимости, и сразу же начать воссоздание компонента. Это пригодится для воссоздания сбояного компонента после его замены.

-s устройство

Выдать информацию о состоянии всех компонентов и резервных дисков указанного **устройства** RAIDframe.

-S устройство

Проверить состояние процессов перезаписи четности, пересоздания компонентов и обратного копирования данных на компоненты. Выдать, насколько зевершен каждый из этих процессов.

-u устройство

Деконфигурировать устройство RAIDframe.

-v

Выдавать более детальную информацию. Для операций вроде пересоздания, перезаписи четности и обратного копирования выдавать индикатор завершенности.

Устройство, используемое утилитой **raidctl**, задается в командной строке. Можно указывать либо полное имя устройства, например, **/dev/rraid0d** для архитектуры i386 и **/dev/rraid0c** для всех остальных, или просто **raid0** (ему будет соответствовать устройство **/dev/rraid0d**).

Формат файла конфигурации - сложный, и здесь представлено лишь краткое его описание. В файлах конфигурации '#' обозначает начало комментария.

В файле конфигурации есть 4 обязательных раздела и 2 необязательных. Каждый раздел начинается со слова '**START**', после которого идет имя раздела и параметры конфигурации этого раздела. Первый раздел - '**array**', задает количество строк, столбцов и резервных дисков в RAID-наборе. Например:

```
START array  
1 3 0
```

задает массив из одной строки, 3 столбцов и 0 резервных дисков. Учтите, что хотя так можно задавать и многомерные массивы, они **НЕ** поддерживаются драйвером.

Второй раздел, '**disks**', задает фактические компоненты устройства. Например, раздел:

```
START disks
/dev/da0s1e
/dev/da1s1e
/dev/da2s1e
```

задает три диска-компоненты для использования в RAID-устройстве. Если любой из указанных дисков неизвестен при конфигурировании RAID-устройства, он будет помечен как сбойный и система будет работать в деградированном режиме. Учтите, что требуется обеспечить неизменность порядка компонентов в файле конфигурации между конфигурациями RAID-устройства. Изменение порядка компонентов приведет к потере данных, если набор конфигурируется с опцией **-C**. Обычно, если задана опция **-c**, RAID-набор не будет конфигурироваться при изменении порядка компонентов.

Следующий раздел, '**spare**', - не обязательный, и, если есть, задает устройства для использования в качестве "горячего резерва". Эти устройства подключены, но не используются активно драйвером RAID пока не произойдет сбой одного из основных компонентов. Простой раздел '**spare**' может иметь вид:

```
START spare
/dev/da3s1e
```

для конфигурации с одним резервным компонентом. Если конфигурация не предполагает использование дисков горячего резерва, раздел '**spare**' можно не задавать.

Следующий раздел - '**layout**'. Он описывает общие параметры организации для RAID-устройства и предоставляет информацию о количестве секторов на полосу, полос на информацию о четности, полос на единицу восстановления и конфигурацию четности, которую надо использовать. Этот раздел может иметь вид:

```
START layout
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level
32 1 1 5
```

Количество секторов на полосу (sectors per stripe) задает коэффициент чередования, в блоках; т.е. количество подряд идущих секторов для записи на каждый компонент в виде полосы. Правильный выбор этого значения (в рассмотренном примере, 32) - тема многих исследований по архитектуре RAID. Количество полос на информацию о четности и полос на единицу восстановления обычно устанавливается равным 1. Хотя определенные значения больше 1 допускаются, обсуждение допустимых значений и последствий использования других значений, кроме 1, выходит за рамки этого документа. Последнее значение в этом разделе (в рассмотренном примере, 5) задает требуемую конфигурацию четности. Допускаются следующие значения:

- | | | |
|---|----------------|--|
| 0 | RAID уровня 0. | Никакой четности, только простое разбиение на полосы. |
| 1 | RAID уровня 1. | Зеркалирование. Информация о четности - просто зеркало. |
| 4 | RAID уровня 4. | Разбиение на полосы по компонентам, с сохранением информации о четности на последнем компоненте. |
| 5 | RAID уровня 5. | Разбиение на полосы по компонентам, с распределением информации о четности по всем компонентам. |

Есть и другие допустимые значения, включая чередование четности (Even-Odd parity), RAID уровня 5 с циклическим резервированием (rotated sparing), цепочной декластеризацией (Chained declustering) и перемежающейся декластеризацией (Interleaved declustering), но на момент написания этого руководства код для этих операций с четностью еще не был протестирован в ОС FreeBSD.

Следующий обязательный раздел - раздел '**queue**'. Чаще всего его задают так:

```
START queue
fifo 100
```

где метод организации очереди задается как **fifo** (первым вошел, первым вышел), а размер очереди для каждого компонента ограничивается 100 запросами. Можно указывать и другие методы организации очереди, но их обсуждение выходит за рамки данного документа.

Последний раздел, '**debug**', - не обязательный. Подробнее о нем см. в документации по **RAIDframe**, которая описана в разделе [ИСТОРИЯ](#).

Пример полного файла конфигурации см. в разделе "[ПРИМЕРЫ](#)".

ПРИМЕРЫ

Системным администраторам настоятельно рекомендуется перед использованием драйвера RAID для реальных производственных систем хорошо ознакомиться с возможностями утилиты **raidctl**, а также понять, как работает процесс восстановления компонентов. Примеры в этом разделе будут посвящены конфигурированию нескольких различных RAID-наборов с разной степенью избыточности. Проработав эти примеры, администраторы смогут хорошо прочувствовать, как конфигурировать RAID-наборы и как инициировать пересоздание вышедших из строя компонентов.

В следующих примерах '**raid0**' будет использоваться для обозначения RAID-устройства. В зависимости от архитектуры, реально надо будет использовать '/dev/rraid0c' или '/dev/rraid0d' вместо '**raid0**'.

Инициализация и конфигурирование

Начальный шаг в конфигурировании RAID-набора - идентифицировать компоненты, которые будут использоваться в RAID-наборе. Все компоненты должны быть одного размера. Каждый компонент должен в **disklabel** иметь тип **FS_RAID**, а типичная запись **disklabel** для компонента **RAID** может иметь вид:

```
f: 1800000200495  RAID      # (Cyl. 405*- 4041*)
```

Хотя **FS_BSDFFS** тоже подходит в качестве типа компонента, тип **FS_RAID** предпочтительнее использовать для механизма RAIDframe, и он является обязательным для поддержки некоторых возможностей вроде автоматического конфигурирования. В процессе начального конфигурирования каждого RAID-набора, каждому компоненту будет задана "метка компонента". "Метка компонента" содержит важную информацию о компоненте, включая заданный пользователем серийный номер, строку и столбец этого компонента в RAID-наборе, уровень избыточности RAID-набора, счетчик изменений ('modification counter') и признак известной корректности информации о четности (если она есть) для компонента. Метки компонентов являются составной частью RAID-набора, поскольку они используются для проверки того, что компоненты сконфигурированы в правильном порядке, а также для хранения другой жизненно важной информации о RAID-наборе. Метки компонентов также необходимы для автоматического выявления и конфигурирования RAID-наборов в ходе загрузки. Чтобы метка компонента считалась допустимой, она должна быть согласована с метками других компонентов в наборе. Например, должны быть согласованы серийный номер, счетчик изменений, количество строк и столбцов. Если любой из этих параметров отличается, то компонент не считается частью набора. Подробнее о метках компонентов см. на странице справочного руководства [raid\(4\)](#).

Как только компоненты идентифицированы и диски имеют соответствующие метки, можно использовать утилиту **raidctl** для конфигурирования устройства **raid(4)**. Чтобы сконфигурировать устройство, необходимо создать файл конфигурации вида:

```
START array
# numRow numCol numSpare
1 3 1

START disks
/dev/da1s1e
/dev/da2s1e
/dev/da3s1e

START spare
/dev/da4s1e

START layout
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level_5
32 1 1 5

START queue
fifo 100
```

Представленный выше файл конфигурации задает набор RAID 5, состоящий из компонентов **/dev/da1s1e**, **/dev/da2s1e** и **/dev/da3s1e**, и использующий **/dev/da4s1e** в качестве устройства "горячего резерва" на случай сбоя одного из трех основных дисков. Набор RAID 0 можно задать аналогично:

```
START array
# numRow numCol numSpare
1 4 0

START disks
/dev/da1s10e
/dev/da1s11e
/dev/da1s12e
/dev/da1s13e

START layout
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level_0
64 1 1 0

START queue
fifo 100
```

В этом случае, устройства **/dev/da1s10e**, **/dev/da1s11e**, **/dev/da1s12e** и **/dev/da1s13e** являются компонентами, образующими этот RAID-набор. Обратите внимание, что устройств горячего резерва в наборе RAID 0 нет, поскольку нет способа восстановить данные при сбое любого из компонентов.

Для набора RAID 1 (зеркалирования) можно использовать следующую конфигурацию:

```
START array
# numRow numCol numSpare
1 2 0

START disks
/dev/da2s10e
/dev/da2s11e

START layout
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level_1
128 1 1 1

START queue
fifo 100
```

В этом случае, **/dev/da2s10e** и **/dev/da2s11e** - два компонента зеркального набора. Хотя в этой конфигурации не заданы компоненты горячего резервирования, их легко можно добавить, так же, как и в рассмотренной выше конфигурации RAID 5. Учтите также, что наборы RAID 1 в настоящее время ограничены только двумя компонентами. Сейчас n-кратное зеркалирование невозможно.

При первом конфигурировании RAID-набора необходимо использовать опцию **-C**:

```
raidctl -C raid0.conf
```

где '**raid0.conf**' - имя файла конфигурации RAID. Опция **-C** обеспечивает успешное конфигурирование, даже если какие-то метки компонентов некорректны. Опцию **-C** не стоит использовать "походя" в ситуациях, не связанных с первоначальным конфигурированием, поскольку если система отказывается конфигурировать RAID-набор, вероятно, тому есть веские причины. После выполнения первоначального конфигурирования (и добавления соответствующих меток компонентов с помощью опции **-I**) обычно устройство **raid0** можно сконфигурировать с помощью команды:

```
raidctl -c raid0.conf
```

Когда RAID-набор конфигурируется впервые, необходимо проинициализировать метки компонентов, а также проинициализировать информацию о четности для RAID-набора. Инициализация меток компонентов выполняется командой:

```
raidctl -I 112341 raid0
```

где '**112341**' - заданный пользователем серийный номер для RAID-набора. Эта инициализация требуется для всех RAID-наборов. Кроме того, настоятельно рекомендуется использование различных серийных номеров для разных RAID-наборов, поскольку использование одного серийного номера для всех RAID-наборов просто уменьшает полезность процедуры проверки меток компонентов.

Инициализация RAID-набора выполняется с помощью опции **-i**. Эта инициализация **должна** быть выполнена для всех RAID-наборов, поскольку, среди прочего, при этом проверяется корректность информации о четности (если она поддерживается) для RAID-набора. Поскольку эта инициализация может требовать достаточно много времени, можно использовать опцию **-v** вместе с **-i**:

```
raidctl -iv raid0
```

Это даст более подробное отображение состояния процесса инициализации:

```
Initiating re-write of parity
Parity Re-write status:
 10% | ****                                | ETA: 06:03 /
```

В результате выдается 'Процент выполнения' в числовом и графическом виде, а также оценочное время до завершения процесса.

Поскольку именно информация о четности обеспечивает "избыточность" в RAID, принципиально важно максимально обеспечить ее корректность. Если информация о четности некорректна, нет гарантии, что данные не будут потеряны при сбое компонента.

Как только известно, что информация о четности корректна, можно безопасно выполнять команды **disklabel(8)**, **newfs(8)** или **fsck(8)** для устройства или его файловых систем, а затем - монтировать файловые системы для использования.

При определенных обстоятельствах (например, дополнительный компонент еще не доставлен или данные надо перенести с диска, который затем станет компонентом массива) было бы желательно сконфигурировать набор RAID 1 с единственным компонентом. Этого можно добиться, конфигурируя набор с физически существующим компонентом (в качестве первого или второго) и "поддельным" компонентом. В следующей конфигурации:

```
START array
# numRow numCol numSpare
1 2 0

START disks
/dev/da6s1e
/dev/da0s1e
```

```

START layout
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level_1
128 1 1 1

START queue
fifo 100

```

/dev/da0s1e - реальный компонент, и будет использоваться как второй диск в наборе RAID 1. Компонент **/dev/da6s1e**, который должен существовать, но не обязательно быть связан с физическим устройством, используется просто в качестве "заместителя". Конфигурирование (с помощью опций **-C** и **-I 12345**, как было показано выше) проходит нормально, но процессу инициализации RAID-набора придется ждать, пока все физические компоненты не будут доступны. После конфигурирования этот набор можно использовать как обычно, но работать он будет в деградированном режиме. Как только будет получен второй физический компонент, его можно будет добавить на ходу, зеркаливать существующие данные и возобновить нормальную работу.

Сопровождение RAID-набора

После первоначальной инициализации информации о четности, команда:

```
raidctl -p raid0
```

может использоваться для проверки текущего состояния информации о четности. Для проверки информации о четности и ее пересоздания при необходимости (например, после нештатной остановки) используется команда:

```
raidctl -P raid0
```

Учтите, что перезапись четности может выполняться одновременно с другими действиями с RAID-массивом (например, пока для файловой системы на RAID-наборе выполняется команда **fsck(8)**). Однако: для максимальной эффективности RAID-набора надо удостовериться в корректности информации о четности до изменения любых данных в наборе.

Чтобы узнать состояние RAID-набора, можно использовать следующую команду:

```
raidctl -s raid0
```

Результат ее выполнения будет иметь вид:

```

Components:
    /dev/dals1e: optimal
    /dev/da2s1e: optimal
    /dev/da3s1e: optimal
Spares:
    /dev/da4s1e: spare
Component label for /dev/dals1e:
    Row: 0 Column: 0 Num Rows: 1 Num Columns: 3
    Version: 2 Serial Number: 13432 Mod Counter: 65
    Clean: No Status: 0
    sectPerSU: 32 SUsPerPU: 1 SUsPerRU: 1
    RAID Level: 5 blocksize: 512 numBlocks: 1799936
    Autoconfig: No
    Last configured as: raid0
Component label for /dev/da2s1e:
    Row: 0 Column: 1 Num Rows: 1 Num Columns: 3
    Version: 2 Serial Number: 13432 Mod Counter: 65
    Clean: No Status: 0
    sectPerSU: 32 SUsPerPU: 1 SUsPerRU: 1
    RAID Level: 5 blocksize: 512 numBlocks: 1799936
    Autoconfig: No
    Last configured as: raid0
Component label for /dev/da3s1e:
    Row: 0 Column: 2 Num Rows: 1 Num Columns: 3
    Version: 2 Serial Number: 13432 Mod Counter: 65
    Clean: No Status: 0
    sectPerSU: 32 SUsPerPU: 1 SUsPerRU: 1

```

```
RAID Level: 5 blocksize: 512 numBlocks: 1799936
Autoconfig: No
Last configured as: raid0
Parity status: clean
Reconstruction is 100% complete.
Parity Re-write is 100% complete.
Copyback is 100% complete.
```

Это показывает, что с RAID-набором все в порядке. Важны здесь строки компонентов со значениями '**'optimal'**', а также строка '**'Parity status'**', свидетельствующая о том, что информация о четности актуальна. Учтите, что если на RAID-наборе есть открытые файловые системы, отдельные компоненты не будут "чистыми" ('**clean**'), но весь набор в целом все равно может быть в состоянии **clean**.

Для проверки метки компонента **/dev/dals1e** используется следующая команда:

```
raidctl -g /dev/dals1e raid0
```

Результат выполнения этой команды будет иметь вид:

```
Component label for /dev/dals1e:
  Row: 0 Column: 0 Num Rows: 1 Num Columns: 3
  Version: 2 Serial Number: 13432 Mod Counter: 65
  Clean: No Status: 0
  sectPerSU: 32 SUsPerPU: 1 SUsPerRU: 1
  RAID Level: 5 blocksize: 512 numBlocks: 1799936
  Autoconfig: No
  Last configured as: raid0
```

Реакция на сбои компонентов

Если по любой причине (например, для тестирования воссоздания) необходимо сымитировать сбой диска, это можно сделать с помощью команды:

```
raidctl -f /dev/da2s1e raid0
```

После этого система будет выполнять все операции в деградированном режиме, с воссозданием недостающих данных на основе существующих и информации о четности. В данном случае, при запросе состояния **raid0** получим (в частности):

```
Components:
  /dev/dals1e: optimal
  /dev/da2s1e: failed
  /dev/da3s1e: optimal
Spares:
  /dev/da4s1e: spare
```

Учтите, что указание опции **-f** не начинает пересоздание. Чтобы пометить диск как сбойный и начать его пересоздание, необходимо указать опцию **-F**:

```
raidctl -F /dev/da2s1e raid0
```

Можно сначала применить опцию **-f**, а потом, позже, опцию **-F**, при желании, для того же диска. Сразу после начала пересоздания при запросе состояния набора будет получено:

```
Components:
  /dev/dals1e: optimal
  /dev/da2s1e: reconstructing
  /dev/da3s1e: optimal
Spares:
  /dev/da4s1e: used_spare
[...]
Parity status: clean
Reconstruction is 10% complete.
Parity Re-write is 100% complete.
Copyback is 100% complete.
```

Это показывает, что пересоздание идет. Чтобы следить за ходом процесса пересоздания можно использовать опцию **-S**. Она будет отображать ход процесса как процент выполненного пересоздания. Когда пересоздание будет закончено, опция **-S** выдаст:

```
Components:  
    /dev/dals1e: optimal  
    /dev/da2s1e: spared  
    /dev/da3s1e: optimal  
Spares:  
    /dev/da4s1e: used_spare  
[...]  
Parity status: clean  
Reconstruction is 100% complete.  
Parity Re-write is 100% complete.  
Copyback is 100% complete.
```

В этот момент, есть, по крайней мере, две возможности. Во-первых, если известно, что устройство **/dev/da2s1e** работает нормально (т.е. сбой был вызван применением опции **-f** или **-F**, или сбойный диск заменен), можно инициировать обратное копирование данных с помощью опции **-B**. В нашем примере это приведет к копированию всего содержимого **/dev/da4s1e** на **/dev/da2s1e**. Когда процедура обратного копирования будет закончена, состояние устройства будет отображаться как (в частности):

```
Components:  
    /dev/dals1e: optimal  
    /dev/da2s1e: optimal  
    /dev/da3s1e: optimal  
Spares:  
    /dev/da4s1e: spare
```

и система снова работает нормально.

Вторая возможность после пересоздания - просто использовать **/dev/da4s1e** вместо **/dev/da2s1e** в файле конфигурации. Например, часть файла конфигурации может иметь вид:

```
START array  
1 3 0  
  
START drives  
/dev/dals1e  
/dev/da4s1e  
/dev/da3s1e
```

Это можно сделать, поскольку в этот момент устройство **/dev/da4s1e** полностью взаимозаменяемо с **/dev/da2s1e**. Учтите, что при изменении порядка устройств в наборе надо быть особенно осторожным. Это - один из немногих случаев, когда устройства и/или их порядок можно поменять без потери данных! В общем случае, порядок указания компонентов в файле конфигурации не должен меняться.

Если происходит сбой компонента и нет подключенных устройств горячего резерва, состояние RAID-набора может быть (в частности) таким:

```
Components:  
    /dev/dals1e: optimal  
    /dev/da2s1e: failed  
    /dev/da3s1e: optimal  
No spares.
```

В этом случае возможны несколько вариантов. Во-первых, можно добавить устройство для горячего резервирования с помощью команды:

```
raidctl -a /dev/da4s1e raid0
```

После такого добавления состояние набора будет таким:

```
Components:  
    /dev/dals1e: optimal  
    /dev/da2s1e: failed  
    /dev/da3s1e: optimal
```

```
Spares:  
        /dev/da4s1e: spare
```

После этого можно выполнить пересоздание с помощью опции **-F**, как было описано выше.

Во-вторых, можно восстановить компонент непосредственно на **/dev/da2s1e**. После замены диска, содержащего компонент **/dev/da2s1e**, можно просто использовать команду:

```
raidctl -R /dev/da2s1e raid0
```

для восстановления компонента **/dev/da2s1e**. По ходу восстановления состояние будет таким:

```
Components:  
        /dev/dals1e: optimal  
        /dev/da2s1e: reconstructing  
        /dev/da3s1e: optimal  
No spares.
```

а после его завершения:

```
Components:  
        /dev/dals1e: optimal  
        /dev/da2s1e: optimal  
        /dev/da3s1e: optimal  
No spares.
```

В случаях, когда определенный компонент полностью недоступен после перезагрузки, для обозначения недостающего компонента может использоваться специальное имя. Например:

```
Components:  
        /dev/da2s1e: optimal  
        component1: failed  
No spares.
```

показывает, что второй компонент этого RAID-набора не был выявлен вообще процедурой автоматического конфигурирования. Имя '**component1**' может использоваться там же, где и имя обычного компонента. Например, чтобы добавить устройство горячего резервирования у указанному выше набору и восстановить данные на это устройство можно сделать следующее:

```
raidctl -a /dev/da3s1e raid0  
raidctl -F component1 raid0
```

после чего данные недостающего '**component1**' будут пересозданы на **/dev/da3s1e**.

RAID на базе RAID

RAID-наборы можно использовать на нескольких уровнях, чтобы создавать более сложные и объемные RAID-наборы. Набор RAID 0, например, может быть создан из четырех наборов RAID 5. Следующий файл конфигурации иллюстрирует такой вариант:

```
START array  
# numRow numCol numSpare  
1 4 0

START disks  
/dev/raid1e  
/dev/raid2e  
/dev/raid3e  
/dev/raid4e

START layout  
# sectPerSU SUsPerParityUnit SUsPerReconUnit RAID_level_0  
128 1 1 0

START queue
```

```
fifo 100
```

Аналогичный файл конфигурации может использоваться для создания набора RAID 0 из компонентов в виде наборов RAID 1. В такой конфигурации зеркалирование обеспечивает высокую степень избыточности, в разбиение на полосы дает ускорение работы.

Автоматическое конфигурирование и размещение корневой файловой системы на RAID

RAID-наборы могут также автоматически конфигурироваться при загрузке. Чтобы сделать набор автоматически конфигурируемым, просто подготовьте RAID-набор, как было описано выше, а затем выполните команду:

```
raidctl -A yes raid0
```

для включения автоматического конфигурирования этого набора. Для отключения автоматического конфигурирования используется команда:

```
raidctl -A no raid0
```

Автоматически конфигурируемые RAID-наборы будут конфигурироваться перед монтированием корневой файловой системы. Эти RAID-наборы, таким образом, доступны для использования в качестве корневой или любой другой файловой системы. Основное преимущество использования автоматического конфигурирования состоит в том, что компоненты RAID становятся менее зависимыми от дисков, на которых они находятся. Например, могут меняться идентификаторы SCSI (SCSI ID's), но автоматически конфигурируемые наборы всегда будут сконфигурированными корректно, даже если идентификаторы SCSI у компонентов поменялись.

Размещение корневой файловой системы (/) на RAID-наборе также допускается, и при этом на таком RAID-наборе для / используется раздел 'a'. Чтобы использовать устройство **raid0a** в качестве корневой файловой системы, просто выполните:

```
raidctl -A root raid0
```

Чтобы сделать устройство **raid0a** только автоматически конфигурируемым, просто используйте параметры **-A yes**.

Учтите, что ядра могут непосредственно читаться только с компонентов RAID 1 на архитектурах **alpha** и **rmax**. На этих архитектурах файловая система **FS_RAID** распознается блоками загрузки, которые могут правильно загружать ядро непосредственно с компонента набора RAID 1. Для других архитектур или для поддержки корневой файловой системы на других RAID-наборах, необходимо использовать какой-то другой механизм для обеспечения загрузки ядра. Например, можно использовать небольшой раздел, содержащий только блоки вторичного загрузчика и альтернативное ядро (или пару ядер). Как только ядро загрузится, однако, оно находит автоматически конфигурируемый RAID-набор, пригодный для использования в качестве корневого, а затем этот набор автоматически конфигурируется и используется как корневое устройство. Если в качестве корневых себя заявляет два или более RAID-набора, пользователю предложат выбрать корневое устройство. В настоящее время в качестве корневого устройства можно использовать наборы RAID 0, 1, 4 и 5.

Типичная структура RAID 1 с корнем на RAID может быть такой:

1. **wd0a** - небольшой раздел, содержащий полную, загружаемую, простейшую установку ОС (*B оригинале - NetBSD, из которой, собственно, RAIDframe в OC FreeBSD и попал. Ну, не успели руководство поменять. Прим. переводчика.*).

2. **wd1a** - также содержит полную, загружаемую, простейшую установку ОС.
3. **wd0e** и **wd1e** - набор RAID 1, **raid0**, используемый для корневой файловой системы.
4. **wd0f** и **wd1f** - набор RAID 1, **raid1**, который будет использоваться только для области подкачки.
5. **wd0g** и **wd1g** - набор RAID 1, **raid2**, используется для **/usr**, **/home** или, при необходимости, для других данных.
6. **wd0h** и **wd0h** - набор RAID 1, **raid3**, если необходимо.

RAID-наборы **raid0**, **raid1** и **raid2** все помечены как автоматически конфигурируемые. **raid0** помечен как используемый для корневой файловой системы. При установке новых ядер, ядро копируется не только в **/**, но также и на устройства **wd0a** и **wd1a**. Ядро на устройстве **wd0a** необходимо, поскольку именно с него и загружается система. Ядро на устройстве **wd1a** также необходимо, поскольку именно оно и будет использоваться при сбое **wd0**. Важно иметь дополнительные копии ядра, на случай сбоя одного из дисков.

Не обязательно размещать корневую файловую систему на том же диске, что и ядро. Например, получение ядра с устройства **wd0a**, и использование **da0s1e** и **da1s1e** для **raid0** и корневой файловой системы, - вполне нормально. Принципиально важно, однако, иметь несколько ядер, на случай сбоя носителя.

Многоуровневые RAID-устройства (такие как набор RAID 0, составленный из наборов RAID 1) не поддерживаются пока в качестве корневых или автоматически конфигурируемых. (Многоуровневые RAID-устройства, в общем случае, однако, поддерживаются, как было сказано ранее.) Учтите, что для включения автоматического конфигурирования RAID-устройств в файле конфигурации ядра должна быть строка:

```
options      RAID_AUTOCONFIG
```

Подробнее об этом см. на странице [raid\(4\)](#).

Деконфигурирование

Последняя операция, выполняемая с помощью **raidctl**, - это деконфигурирование устройства [raid\(4\)](#). Это делается с помощью простой команды:

```
raidctl -u raid0
```

после чего устройство готово для переконфигурирования.

Настройка производительности

Подбор значений различных параметров для максимальной производительности может быть делом весьма сложным, и часто требует использования метода проб и ошибок для подбора наиболее подходящих для конкретной системы параметров. В игру вступает целый ряд факторов:

1. Типы компонентов (SCSI или IDE) и их пропускная способность
2. Типы контроллеров и их пропускная способность
3. Распределение компонентов по контроллерам
4. Пропускная способность ввода-вывода
5. Особенности доступа к файловой системе
6. Скорость процессора

Как и при любой настройке производительности, оценка производительности при реальных нагрузках может оказаться единственным способом оценки предполагаемой производительности. Понимание хотя бы части базовых технологий тоже не повредит при настройке. Цель этого раздела - указать те параметры, которые могут существенно влиять на производительность.

Для набора RAID 1, значение **SectPerSU**, равное 64 или 128, обычно подойдет. Поскольку данные в наборе RAID 1 организованы в каждом компоненте линейно, выбор подходящего размера полосы менее критичен, чем для набора RAID 5. Однако слишком маленький размер полосы будет приводить к разбиению большого запроса ввода-вывода на несколько маленьких, что отрицательно скажется на производительности. В то же время, большой размер полосы может вызвать проблемы с одновременным доступом к полосам, что тоже может снизить производительность. Поэтому значения в диапазоне от 32 до 128 обычно наиболее эффективны.

Настройка наборов RAID 5 сложнее. В самом лучшем случае, запросы ввода-вывода будут поступать на RAID-набор по одной полосе за раз. Поскольку в начале ввода-вывода доступна вся полоса, четность для нее можно вычислить до записи, а затем данные и информацию о четности можно записывать параллельно. Когда размер записываемых данных меньше, чем размер полосы, возникает проблема 'меленькой записи'. Поскольку 'меленькая запись' означает, что меняется будет только часть полосы на компонентах, данные (и информация о четности) на компонентах должны меняться несколько иначе. Во-первых, 'старую четность' и 'старые данные' надо прочитать с компонентов. Затем строится новая информация о четности, используя новые данные, которые надо записать, старые данные и прежнюю информацию о четности. Наконец, новые данные и новая информация о четности записываются. Все эти дополнительные манипуляции с данными приводят к серьезному снижению производительности, и такая запись обычно выполняется в 2-4 раза медленнее, чем запись (или чтение) полной полосы. Чтобы справиться с этой проблемой в реальной ситуации, может пригодиться задание достаточно маленьких полос, чтобы "большой запрос ввода-вывода" от системы использовал ровно одну большую запись полосы. Как оказалось в дальнейшем, определенную роль тут могут сыграть также особенности файловой системы.

Поскольку размер "большого запроса ввода-вывода" часто (в настоящее время) составляет 32 или 64 Кбайта, на пятидисковом наборе RAID 5 желательными для параметра **SectPerSU** могут оказаться значения 16 блоков (8 Кбайт) или 32 блока (16 Кбайт). Поскольку имеется 4 сектора данных на полосу, максимальный объем данных в полосе будет 64 блока (32 Кбайта) или 128 блоков (64 Кбайта). И в этом случае только экспериментальные оценки позволят выяснить, какие значения дают лучшую производительность.

Параметры, используемые при создании файловой системы, также принципиально влияют на производительность. Для **newfs(8)**, например, увеличения размера блока до 32 Кбайт или 64 Кбайт может существенно повысить производительность. Кроме того, изменение параметра, задающего количество

цилиндров в группе, с 16 на 32 или более часто не только необходимо для поддержки больших файловых систем, но и может положительно повлиять на производительность.

Резюме

Несмотря на размер этой страницы справочного руководства, конфигурирование RAID-набора - процесс относительно простой. Необходимо выполнить следующие шаги:

1. С помощью [disklabel\(8\)](#) создать компоненты (типа **RAID**).

2. Создать файл конфигурации RAID, например, '**raid0.conf**

3. Сконфигурировать RAID-набор:

4. `raidctl -C raid0.conf`

5. Проинициализировать метки компонентов:

6. `raidctl -I 123456 raid0`

7. Проинициализировать другие важные части набора:

8. `raidctl -i raid0`

9. Получить стандартную метку для RAID-набора:

10. `disklabel raid0 > /tmp/label`

11. Отредактировать метку:

12. `vi /tmp/label`

13. Записать новую метку на RAID-набор:

14. `disklabel -R -r raid0 /tmp/label`

15. Создать файловую систему:

16. `newfs /dev/rraid0e`

17. Смонтировать файловую систему:

18. `mount /dev/raid0e /mnt`

19. Используйте команду:

20. `raidctl -c raid0.conf`

для переконфигурирования RAID-набора в следующий раз, когда он понадобится, или поместите файл **raid0.conf** в каталог **/etc**, и тогда массив будет автоматически запускаться сценариями **/etc/rc**.

ПРЕДУПРЕЖДЕНИЯ

Некоторые уровни RAID (1, 4, 5, 6 и другие) могут защитить от потери данных при сбое одного компонента. Однако потеря двух компонентов системы RAID 4 или 5, либо потеря одного компонента системы RAID 0 приводит к потере всей файловой системе на этом RAID-устройстве. RAID - не замена продуманной стратегии резервного копирования.

Перевычисление четности **должно** выполняться в любой ситуации, когда есть шанс, что эти данные могут быть повреждены. К этим ситуациям относятся сбои системы или добавление ранее не использовавшегося RAID-устройства. Некорректная информация о четности приведет к катастрофе в случае любого сбоя компонента - лучше использовать RAID 0, и получить дополнительное место и выигрыш в скорости, чем хранить информацию о четности, но не обеспечивать ее корректность. Использование RAID 0 хотя бы не создает видимость повышенной защиты данных.

ФАЙЛЫ

/dev/{r}raid*
специальные файлы устройств raid.

ССЫЛКИ

[ccd\(4\)](#), [raid\(4\)](#), [rc\(8\)](#)

ОШИБКИ

Удаление диска горячей замены пока невозможно.

ИСТОРИЯ

RAIDframe - это механизм для быстрого прототипирования RAID-структур, разработанный сотрудниками Parallel Data Laboratory университета Карнеги-Мэллона (Carnegie Mellon University - CMU). Более полное описание внутреннего устройства и возможностей RAIDframe можно найти в статье "**RAIDframe: A Rapid Prototyping Tool for RAID Systems**", by William V. Courtright II, Garth Gibson, Mark Holland, LeAnn Neal Reilly, and Jim Zelenka, опубликованной Parallel Data Laboratory университета Карнеги-Мэллона.

Команда **raidctl** впервые появилась как отдельная программа в дистрибутиве RAIDframe v1.1 CMU. Данная версия **raidctl** полностью переписана и впервые появилась в OC FreeBSD 4.4.

АВТОРСКИЕ ПРАВА

Авторские права на RAIDframe (RAIDframe Copyright) следующие:

Copyright (c) 1994-1996 Carnegie-Mellon University.
All rights reserved.

Permission to use, copy, modify and distribute this software and its documentation is hereby granted, provided that both the copyright notice and this permission notice appear in all copies of the software, derivative works or modified versions, and any portions thereof, and that both notices appear in supporting documentation.

CARNEGIE MELLON ALLOWS FREE USE OF THIS SOFTWARE IN ITS "AS IS" CONDITION. CARNEGIE MELLON DISCLAIMS ANY LIABILITY OF ANY KIND FOR ANY DAMAGES WHATSOEVER RESULTING FROM THE USE OF THIS SOFTWARE.

Университет Carnegie Mellon требует от пользователей этого ПО пересыпать по адресу:

Software Distribution Coordinator или Software.Distribution@CS.CMU.EDU
School of Computer Science
Carnegie Mellon University
Pittsburgh PA 15213-3890

любые улучшения или расширения, которые они сделали, и предоставлять университету права на дальнейшее распространение этих изменений.

FreeBSD 4.9, 6 ноября 1998 года

Copyleft (no c) - Fuck copyright! 2004 [В.Кравчук](#), [OpenXS Initiative](#), перевод на русский язык