Перенос системного раздела на RAID

Ситуация: продакшн-сервер, один диск, что с бэкапами - неизвестно.

Задача: добавить в систему два новых диска, объединить их в RAID-1 (зеркало), перенести на них систему со всеми потрохами с минимальным даунтаймом, старый диск использовать под бэкапы. Железного RAID нет, только fake raid, поэтому выкручиваемся при помощи mdadm.

Система: Ubuntu Server 12.04 LTS

Диски:

- Основной, с рабочей системой - sda, один раздел sda1, система ext4
- Новый диск №1 - sdb
- Новый диск №2 - sdc

До начала операции

Обязательно бэкапим все нужное.

Обновляем систему и устанавливаем необходимые пакеты

```
$ sudo apt-get update && sudo apt-get upgrade
$ sudo apt-get install mdadm rsync
```

Добавляем диски в систему

Если сервер поддерживает hot spare - мы на коне. Если нет - можно подключить через USB или иным способом, но в случае большого количества и/или объема данных процесс будет крайне длительным и время простоя будет оочень большим. Не подходит ни один из вариантов? Выключаем сервер и добавляем диски добрым старым способом.

Проверяем все ли диски увиделись:

```
$ sudo fdisk -l | grep Disk
```

Убедимся что текущий системный диск это sda:

```
$ df -h | grep -Po "/dev/..."
```

Создаем файловые системы

Диск №1:

```
$ sudo fdisk /dev/sdb
p (убедимся что диск пуст)
n (создаем новый раздел)
p (тип: первичный)
1 (номер партиции)
[Enter]
[Enter] (все пространство под наш новый раздел)
t (тип раздела)
fd (Linux raid auto)
w (записываем изменения и выходим)
```

Повторяем то же самое для диска №2:

```
$ sudo fdisk /dev/sdc
p (убедимся что диск пуст)
n (создаем новый раздел)
p (тип: первичный)
1 (номер партиции)
[Enter]
[Enter] (все пространство под наш новый раздел)
t (тип раздела)
fd (Linux raid auto)
w (записываем изменения и выходим)
```

Создаем RAID-1 (зеркало)

```
$ sudo mdadm --create /dev/md0 --level=1 --raid-devices=2 /dev/sdb1
/dev/sdc1
```

Какое-то время будет создаваться рейд, в этом можно убедиться глядя на индикаторы обращения к дискам. Посмотреть текущий статус можно так:

```
$ sudo mdadm -D /dev/md0
```

Добавляем информацию о новом массиве в конфигурационный файл:

```
$ sudo mdadm --examine --scan >> /etc/mdadm/mdadm.conf
```

Проверяем название массива - оно должно быть прописано правильно! В противном случае мы получим так называемую проблему «md127» (массив виден, но его имя /dev/md127).

Создаем разделы

Смотрим какие разделы были созданы на основном диске:

```
$ sudo fdisk -l /dev/sda
```

и создаем такие же на новом RAID-массиве:

```
$ sudo fdisk /dev/md0
```

Размер выбираем по желанию.

Не забываем сделать первый раздел загрузочным!

Создаем файловые системы

У меня один общий раздел / и один раздел swap, так что в этом случае все просто:

```
$ sudo mkfs.ext4 /dev/md0p1
$ sudo mkswap /dev/md0p5
```

Переносим файловую систему

Примечание: копирование идет пофайлово, поэтому если у вас работают какие-то сервисы и вносятся изменения в данные (СУБД, веб-сервер и т.д.) необходимо в дальнейшем перед перезагрузкой остановить эти сервисы и сделать выборочное копирование информации или решить вопрос при помощи импорта/экспорта. В противном случае есть шансы потерять какие-либо данные.

```
$ sudo mount /dev/md0p1 /mnt
$ sudo rsync -axu / /mnt/
```

Если есть желание следить за происходящим, нужно добавить **–progress** в команду rsync

Подключаем системные каталоги, входим в новую систему

Теперь необходимо сделать настройки в новой системе, при этом не трогая оригинал. Для этого подключаем системные каталоги и переходим в новую систему.

```
$ sudo mount --bind /proc /mnt/proc
$ sudo mount --bind /dev /mnt/dev
$ sudo mount --bind /sys /mnt/sys
$ sudo mount --bind /run /mnt/run
$ sudo chroot /mnt
```

Последняя команда (chroot) переключает нас в новую копию системы, все остальные операции мы будем делать там. Это позволит сохранить рабочий диск нетронутым и при факапе вновь с него загружаться и работать.

Вносим изменения в fstab

Чтобы система смогла загрузиться из нового места ей необходимо объяснить где у нас будут устройства (диски).

```
/# ls -l /dev/disk/by-uuid | grep md >> /etc/fstab
/# nano -w /etc/fstab
```

Теперь правим fstab: закомментариваем старые UUID, добавляем вместо них новые. Крайне желательно не ошибаться, иначе система просто не загрузится.

Настраиваем и устанавливаем Grub

Чтобы загружаться с нового массива нам нужно а)установить загрузчик и б)правильно его сконфигурировать.

Сначала обновляем конфигурацию:

```
/# update-grub
/# cat /boot/grub/grub.cfg | grep UUID_нового_системного_раздела
```

Необходимо убедиться что новый UUID есть в обновленном конфигурационном файле!

Теперь пора установить загрузчик на наши новые диски:

```
/# grub-install /dev/sdb
```

Обновляем конфигурацию модулей ядра

Это нужно сделать для нормальной работы mdadm, иначе получим вышеописанную проблему «md127»

```
/# update-initramfs -u
```

Выходим в реальную систему

```
/# exit
```

Восстановление рабочих данных

Наступило время восстановления рабочих данных - СУБД и прочее. Но тут каждый все делает самостоятельно.

Перезагрузка в новую систему

```
$ sudo reboot
```

Перезагружаем систему, выбираем для загрузки новые диски (через BIOS или любым другим способом), загружаемся. По идее все должно быть хорошо. На всякий случай проверяем с какого же устройства мы загрузились

```
$ df -h | grep -Po "/dev/..."
/dev/md0
```

Ага, все хорошо, все заработало.

Заключение

Осталось протестировать работу системы и придумать что сделать с освободившимся диском: пустить его в hot swap, под хранение данных или просто выкинуть.

ubuntu, server, migration, production, raid, mdadm, миграция системы, миграция системного раздела