

Клонирование разделов диска в FreeBSD



Владимир Ляшко

Ситуация, когда нужно клонировать разделы диска, – не редкость. Такая копия может понадобиться для восстановления, быстрого развертывания системы. Не лишней она будет и при обучении. К сожалению, в FreeBSD нет удобных программ и понятного алгоритма создания такой копии. Специализированные инструменты вроде Acronis TrueImage или NortonGhost, при помощи которых легко можно создать копию раздела Windows и Linux, с файловой системой UFS не работают. Чтение документации и поиск в Интернете показали, что в FreeBSD для этих целей с успехом используются утилиты `dump/restore`, собственно для этих целей они и предназначены.

С их помощью можно сделать слепок файловой системы разделов. Познакомившись с несколькими руководствами по их работе, сделал вывод, что новичку понять и осилить с ходу алгоритм их работы будет сложно, да и неудачные эксперименты могут дорого стоить. После нескольких опытов на практике удалось найти наиболее простой и понятный путь создания копий разделов и их восстановления в случае отказа жесткого диска. Плюс этот метод срабатывает и в том случае, когда понадобится перенести систему на жесткий диск большего размера, для увеличения размеров определенных разделов диска.

Порядок создания имиджа и восстановления файловой системы опишу на примере рабочего шлюза с установленной FreeBSD 6.3. Перед началом создания имиджа рекомендую сохранить в файл вывод команды «df», чтобы впоследствии не было путаницы с наименованиями партиций при восстановлении, да и просто для того чтобы вспомнить, какие там были разделы вообще.

Итак, имеем один слайс и четыре раздела:

```
# df -h -t ufs
```

Filesystem	Size	Used	Avail	Capacity	Mounted on
/dev/ad0s1a	533M	85M	405M	17%	/
/dev/ad0s1d	1.9G	4.1M	1.7G	0%	/tmp
/dev/ad0s1f	65G	15G	3.2G	95%	/usr
/dev/ad0s1e	4.7G	157M	4.2G	4%	/var

Полученные файлы дампов будем сохранять в расширенную и примонтированную папку Windows компьютера, по протоколу SMB, хотя это потребует большего времени, чем копирование на раздел жесткого диска, и увеличивает нагрузку на сеть, но в моем случае это не критично, ну и не забываем, если это раздел FAT32, то максимальный размер файла, который мы можем записать, равен 4 Гб.

```
# mount_smbfs //turbina/dump /mnt
```

Создаем дамп корневого раздела:

```
# dump -0uL -f /mnt/dump.root /
```

В итоге через некоторое время получим файл dump.root, содержащий копию раздела /dev/ad0s1a, смонтированно как корень.

Значение использованных ключей следующее:

- **0** – полное резервное копирование, гарантирует, что вся файловая система будет скопирована. 0-9 – уровни сохранения. На уровне 0, полное резервное копирование, гарантирует, что вся файловая система будет скопирована. Уровни выше 0, инкрементальное копирование, говорят dump копировать все файлы, новые или модифицированные, с момента последнего сохранения любого уровня. По умолчанию уровень 0.
- **u** – обновляет файл dumpdates после успешного сохранения. Формат файла dumpdates читаем для человека, содержит одну из трех форматов записи на строку: имя файловой системы, инкрементальный уровень и дату сохранения в формате ctime(3). Может быть только одна запись для файловой системы каждого уровня. Файл dumpdates может быть отредактирован для из-

менения любого из полей, если это необходимо. Путь по умолчанию для файла dumpdates – /etc/dumpdates, но опция -D может использоваться для изменения пути.

- **L** – эта опция предупреждает dump, что происходит процесс сохранения живой файловой системы. Для получения полного образа dump записывает снимок файловой системы в директорию .snap в корне файловой системы, которая будет сохранена, и затем делает сохранение снимка. Снимок разлинковывается, как только начинается процесс сохранения, и удаляется после завершения сохранения. Опция игнорируется для отмонтированных или смонтированных только на чтение файловых систем. Если директорию .snap отсутствует в корневой директории сохраняемой файловой системы, будет показано предупреждение, и dump вернется к стандартному поведению. Эта проблема может быть устранена созданием .snap директории, в корневом каталоге файловой системы, которая будет сохраняться; ее владельцем должен быть «root», группа «operator», и атрибуты должны быть «0770».

- **f** – file (файл) Запись бэкапа в файл; файл может быть специальным устройством, таким как /dev/sa0 (ленточный накопитель), /dev/fd1 (дисковод), обычный файл, или «-» (стандартный вывод). Множественные имена файлов могут быть заданы одним аргументом, разделенным запятыми. Каждый файл будет использоваться для одного тома в порядке очереди; если для сохранения требуется больше томов, чем заданных имен, последнее имя будет использовано для всех следующих томов после приглашения для смены носителя. Если имя файла имеет форму: «хост:файл», или «пользователь@хост:файл», dump запишет названный файл на удаленный хост, используя rmt(8). Имя пути по умолчанию для удаленной программы rmt(8) /etc/rmt; путь можно изменить переменной окружения RMT.

В моем случае этих ключей достаточно, об остальных читайте man dump.

Таким же образом создаем дампы и с других разделов:

```
# dump -0uL -f /mnt/dump.var /var
# dump -0uL -f /mnt/dump.tmp /tmp
# dump -0uL -f /mnt/dump.usr /usr
```

В случае создания дампа непримонтированного раздела, или если копируемая ФС не описана в /etc/fstab, используем имя файловой системы (партиции) например:

```
# dump -0uaL -f /mnt/dump.var /dev/ad0s1e
```

В итоге в сетевой папке у нас появится четыре больших файла, по одному на каждый раздел, если вы уверены, что в разделе /tmp рабочей системы ничего важного для вас нет, его можно не бэкапить. В рабочей системе, после создания полного дампа, можно периодически (через cron) делать инкрементные копии разделов /var и /tmp (и остальных, если в них производились изменения), заменив цифру «0» в команде на большую (от 1 до 9 по порядку), это во-первых удобно, а во-вторых значительно снижает нагрузку на дис-

ковую подсистему во время создания бэкапа, чего не скажешь при работе утилиты dump по 0-му уровню дампа.

Порядок восстановления системы на жесткий диск

Есть несколько вариантов первичной подготовки диска. Я выбрал самый простой. Используя первый загрузочный диск FreeBSD, в sysinstall создаем слайс и нужные нам разделы на диске, записываем изменения клавишей W, в следующем меню прописываем загрузчик.

Более продвинутый способ – сделать все это с помощью Live CD Frenzy [2], используя утилиты fdisk, bsdlable, и прописать загрузчик командой:

```
#fdisk -B ad0
```

Затем после перезагрузки компьютера, загружаюсь уже с Live CD Frenzy [2], используя в приглашении boot параметры nohdmnt, hdrw, и проверяем разделы на наличие ошибок:

```
# fsck /dev/ad0s1a
# fsck /dev/ad0s1d
# fsck /dev/ad0s1e
# fsck /dev/ad0s1f
```

При помощи ifconfig настраиваем сеть:

```
# ifconfig rl0 inet 192.168.1.161/24
```

И монтируем раздел с файлами дампов:

```
# mount_smbfs //turbina/dump /mnt/smb
```

Для удобства и чтобы не напутать, что куда монтировать, создаем каталог, в который будем монтировать разделы и восстанавливать файловые системы в следующем порядке: /, /tmp, /var, /usr.

```
# mkdir /home/restore
```

Монтируем корневой раздел /dev/ad0s1a и переходим в него для восстановления корневой файловой системы:

```
# mount -t ufs /dev/ad0s1a /home/restore
# cd /home/restore
```

Восстанавливаем корневой раздел.

```
# restore -vrf /mnt/smb/dump.root
```

Значение использованных ключей следующее:

- **-v (verbose)** – заставляет выводить имя каждого файла.
- **-r (restore)** – восстановление (создание заново файловой системы). Целевая файловая система, должна быть сделана ранее с newfs(8), смонтирована, и пользователь должен перейти в подготовленную ранее файловую систему, прежде чем начнется восстановление нулевого уровня из резервной копии. Если уровень 0 восстановлен удачно, флаг -r может использоваться для восстановления всех необходимых инкрементальных бэкапов выше уровня 0. Флаг -r устраняет интерактивное извлечение файлов и может являться вредным для здоровья, если он не будет применяться аккуратно.
- **-f (file)** – читает бэкап из файла; файл может являться специальным файлом устройства, как например /dev/sa0 (ленточный накопитель), /dev/da1c (дисковый накопитель), простым файлом, или «-» (стандартный вывод). Если имя файла имеет форму: «хост:файл», или «пользователь@хост:файл», restore прочтет названный файл на удаленном хосте, используя rmt(8).

Для проверки целостности носителя резервной копии существует ключ -N, при котором происходит извлечение файлов, но ничего на диск не записывается.


Если возникают какие-то нюансы, значение остальных ключей читайте в man restore.

Аналогичным образом поступаем и с остальными разделами.

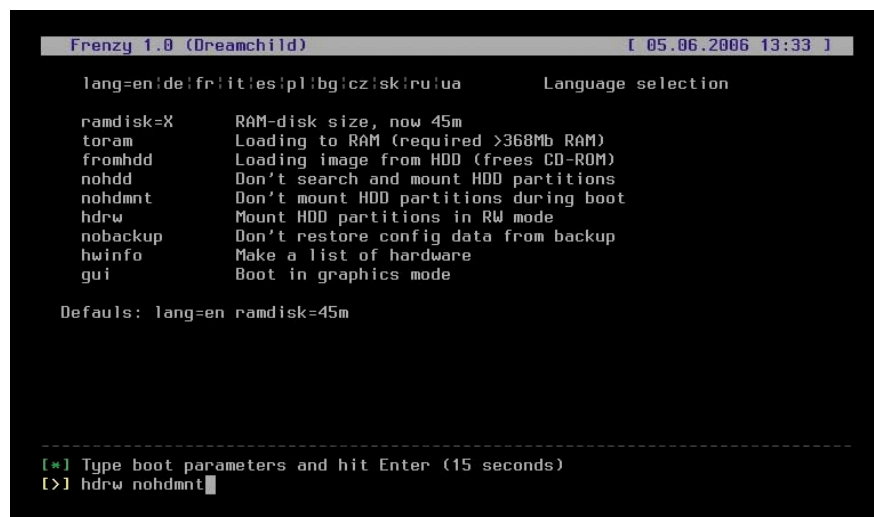
```
# mount -t ufs /dev/ad0s1d /home/restore/tmp
# cd /home/restore/tmp
# restore -vrf /mnt/smb/dump.tmp
# mount -t ufs /dev/ad0s1d /home/restore/var
# cd /home/restore/var
# restore -vrf /mnt/smb/dump.var
# mount -t ufs /dev/ad0s1d /home/restore/usr
# cd /home/restore/usr
# restore -vrf /mnt/smb/dump.usr
```

Размонтируем все разделы в обратном порядке:

```
# umount /home/restore/usr
# umount /home/restore/var
# umount /home/restore/tmp
# umount /home/restore
# umount /mnt/smb
```

Все готово. Перегружаем систему, вынимаем диск. Если все грузится и работает – значит все сделано правильно. Я использую данную схему довольно давно, и пока проблем не было. 

1. Перевод man dump – <http://www.lissyara.su/?id=1481>.
2. Сайт проекта Frenzy – <http://frenzy.org.ua>.



Меню загрузки