Архитектура сетей и систем телекоммуникаций.

Практическая работа 3.

# Работа с разделами жесткого диска и файловой системой в ОС UNIX.

### Цель работы:

- освоить принципы разметки дисков в консоли;

- приобрести навыки использования интерфейса командной строки и простейших команд управление файловой системой, включением и отключением раздела подкачки;

**1. Теория**

**1.1. MBR**

Как известно, перед тем как создавать дисковые разделы для последующего форматирования на нем нужно создать таблицу разделов. Самые распространённые таблицы разделов MBR и GUID. В задачу таблицы входит хранение информации о разметки дисковых разделов. То есть информация о том, где начинается, заканчивается один дисковый том и начинается другой, а так же хранение такой информации как например атрибут активности диска.

Нулевой сектор физического жёсткого диска содержит так называемую главную загрузочную запись (MBR — Master Boot Record). При начальной загрузке компьютера с жёсткого диска BIOS считывает этот сектор в оперативную память и передаёт управление содержащемуся в нём коду начального загрузчика; подробнее об этом можно прочитать в статье Начальная загрузка компьютера.

MBR логически можно разделить на три области: код начального загрузчика (446 байт), таблицу разделов (четыре записи по 16 байт каждая, первая запись находится по смещению 1BEh от начала MBR) и сигнатуру — слово со значением AA55h, занимающее последние два байта MBR (смещение 1FEh). BIOS проверяет сигнатуру, чтобы убедиться в корректности MBR; если сигнатура не равна указанному значению, загрузка не выполне­ния и выдаётся сообщение об ошибке.

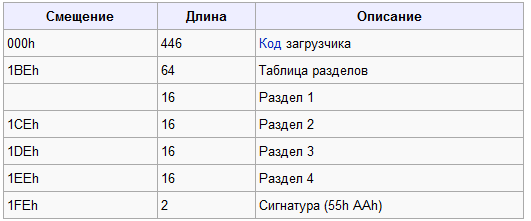


Рисунок 2. Описание содержания MBR

***Код загрузчика***

После завершения процедуры POST в ОЗУ по адресу 0x0000:0x7c00 записывается код загрузчика после чего ему передается управление. Задача загрузчика проанализиро­вать таблицу разделов жёсткого диска, а затем либо передать управление загрузочному коду активного раздела, либо загрузить в память ядро операционной системы и передать ему управление.

***Таблица разделов***

В таблице разделов хранится информация о типе раздела и его расположении на жёстком диске.

***Сигнатура***

Последние два байта MBR называются сигнатурой. Значение этих байтов должно быть 55h AAh. В случае, если это не так, запись считается некорректной.

***Признак активности разделов***

Признак активности раздела — признак обозначающий возможность загрузки операционной системы с данного раздела. Для стандартных загрузчиков может прини­мать следующие значения:

- 80h — раздел является активным

- 00h — раздел является неактивным

- Другие значения являются ошибочными и игнорируются.

Исходя из описания MBR можно выделить главные особенности при создании разделов на HDD:

* Максимальное количество основных разделов **4**
* Для создания логических дисков нужно использовать 1 основной раздел, причем же­лательно последний, так как если создать логические разделы сразу после пер­вого основного раздела – дальнейшие создание 3-го и 4-го разделов будет невоз­можно.
* MBR можно просто резервировать и восстанавливать используя кроссплатформен­ные утилиты для низкоуровневой работы с HDD: fdisk, dd и testdisk.

В лабораторной работе будут применяться утилиты fdisk и dd

**1.2. Утилита FDISK**

Работа с программой fdisk осуществляется через командный интерфейс. Если запустить ее без опций, мы увидим только приглашение к вводу одной из доступных однобуквенных клавиатурных команд:

[root]# /sbin/fdisk /dev/hda

The number of cylinders for this disk is set to 2498.

There is nothing wrong with that, but this is larger than 1024,

and could in certain setups cause problems with:

1) software that runs at boot time (e.g., old versions of LILO)

2) booting and partitioning software from other OSs

(e.g., DOS FDISK, OS/2 FDISK)

Command (m for help):

Список доступных команд мы получаем по команде <m>

Command (m for help): m

Command action

a toggle a bootable flag

b edit bsd disklabel

c toggle the dos compatibility flag

d delete a partition

l list known partition types

m print this menu

n add a new partition

o create a new empty DOS partition table

p print the partition table

q quit without saving changes

s create a new empty Sun disklabel

t change a partition's system id

u change display/entry units

v verify the partition table

w write table to disk and exit

x extra functionality (experts only)

Command (m for help):

Самая употребительная из этих команд, пожалуй, команда <p> по которой выводится полный перечень имеющихся на диске разделов (включая логические). Результат выполнения этой команды аналогичен результату, получаемому после выполнения команды

[root]# /sbin/fdisk -l /dev/hda :

[root]# /sbin/fdisk -l /dev/hda

Disk /dev/hda: 40.0 GB, 40020664320 bytes

16 heads, 63 sectors/track, 77545 cylinders

Units = cylinders of 1008 \* 512 = 516096 bytes

Device Boot Start End Blocks Id System

/dev/hda1 1 765 385528 82 Linux swap

Partition 1 does not end on cylinder boundary.

/dev/hda2 \* 766 4575 1919736 c W95 FAT32 (LBA)

Partition 2 does not end on cylinder boundary.

/dev/hda3 4575 4766 96390 83 Linux

Partition 3 does not end on cylinder boundary.

/dev/hda4 4766 77536 36676395 f W95 Ext'd (LBA)

Partition 4 does not end on cylinder boundary.

/dev/hda5 4766 6806 1028128+ 83 Linux

/dev/hda6 6806 10886 2056288+ 83 Linux

/dev/hda7 10886 27142 8193118+ 83 Linux

/dev/hda8 27142 36896 4915858+ 83 Linux

/dev/hda9 36896 77536 20482843+ b W95 FAT32

Границы разделов в этом случае показаны в виде номеров цилиндров. Вообще говоря, в процессе знакомства с программами разбиения диска складывается впечатление, что границы разделов лучше всего размещать по границам цилиндров.

Если добавить в вызов утилиты fdisk опцию -u, то границы разделов будут задаваться порядковыми номерами секторов. Я предполагаю (хотя нигде в документации такого утверждения я не встречал), что это соответствует линейной нумерации секторов (LBA). Номера секторов вы увидите и тогда, когда выполните команду <p> после запуска самой утилиты в следующем формате:

[root]# /sbin/fdisk -u /dev/hda

Впрочем, переключиться с одного способа указания границ разделов на другой можно в любой момент работы с программой, достаточно дать клавиатурную команду <u>.

Назначение некоторых клавиатурных команд я поясню так же кратко, как они даются самой утилитой, но на русском языке:

**<a>** - снять/установить флаг активности раздела (номер раздела будет запрошен);

**<d>** - удалить раздел;

**<l>** - вывести список известных программе типов разделов;

**<n>** - создать (добавить) новый раздел;

**<o>** - создать новую пустую таблицу разделов в стиле DOS;

**<q>** - выйти из программы, не записывая на диск сделанные изменения;

**<t>** - изменить метку типа раздела (можно задавать один из типов, перечисленных в выводе команды <l>, причем список этот может быть выведен на экран и после вызова команды <t>);

<v> - проверить таблицу разделов (мне показалось, что эта команда мало полезна, но я могу и ошибаться);

<w> - записать исправленную таблицу разделов на диск и выйти из программы;

<x> - дополнительные возможности (только для экспертов).

Две последних команды требуют дополнительных пояснений. Как видите, разработчики программы предусмотрели некие меры предосторожности, чтобы защититься от необдуманных действий пользователя. Пока вы не выполнили команду <w>, внесенные вами изменения еще не являются необратимыми. Вы всегда можете нажать клавишу <q> и таблица разделов останется нетронутой. Вероятно, программа работает с каким-то временным файлом - образом MBR, и только по команде <w> происходит запись содержимого этого файла в MBR.

По команде <x> мы переходим “на второй уровень” программы fdisk. Как сказано выше, этот уровень - для экспертов! Перечень дополнительных возможностей, доступных на этом уровне, можно снова получить по команде <m>

**Утилита dd**

Команда dd практически ровесник ОС Юникс. Днем рождения последней считается 1 января 1970, и точно известно, что уже в 1970 году утилита dd работала с ленточными накопителями, при помощи которых данные переносили с одной ЭВМ на другую, а также запускали и устанавливали ОС Юникс на популярные тогда мини-ЭВМ PDP/11.

**Обычные варианты применения команды dd**

Начнем знакомство с операндами команды dd:

**if=filename (input file)** Этот операнд задает входной файл; если он не указан, то по умолчанию используется стандартный ввод. Этим файлом может быть также файл (нода) устройства, например /dev/hda1, или специфические файлы типа /dev/zero.

**of=filename (output file)** Задает выходной файл; если он не указан, то по умолчанию используется стандартный вывод (экран монитора).

Знакомство с этими двумя операндами уже дает нам возможность использовать программу dd для копирования файлов.

Пример 1. Простое копирование.

# dd if=/home/ya/Desktop/shema.txt of=/home/ya/Desktop/shema.html

3+1 записей считано

3+1 записей написано

скопировано 1549 байт (1,5 kB), 0,427457 секунд, 3,6 kB/s

Как мы видим, программа dd сняла копию файла shema.txt и записала данные в файл shema.html. При этом она выдала нам сообщение.

Разберемся, о чем говорится в этом сообщении. Команда dd считывает и записывает блоками по N байт в каждом. Поскольку мы не задали в командной строке размер блока, то программа использовала размер блока по умолчанию, равный 512 байт. Всего скопировано 1549 байт, как записано в последней строке сообщения. "3+1 записей считано" означает число полных блоков - 3 плюс один неполный блок, содержащий оставшиеся байты 1549-(3\*512)=13. То же относится и к строке "3+1 записей написано".

Теперь самое время разобраться с размером блока.

Сразу скажу, что блок команды dd не имеет никакого отношения к блокам данных файловой системы. Программа dd работает с необработанными "сырыми" (raw data) данными на низком уровне, т.е. на уровне секторов жесткого диска. А любая файловая система является надстройкой над этим уровнем. Правильнее было бы назвать блок буфером. Мы имеем два буфера - входной буфер и выходной буфер. Величина буфера задается операндом bs (block size) и исчисляется в байтах. Например, запись bs=512 означает, что нами установлен размер блока (буфера) в 512 байт. Кстати это значение и используется по умолчанию. Если величина блока не указана, dd использует блоки по 512 байт, что подходит для абсолютного большинства задач. Итак, величина буфера задана, dd создает 512-байтный буфер считывания, посылает единственный запрос на чтение со входного файла, затем создает 512-байтный буфер записи и посылает единственный запрос на запись в выходной файл. Давайте сравним работу программы dd и программы cp при копировании дискеты.

# dd if=/dev/fd0 of=floppy.img bs=1474560

1+0 записей считано

1+0 записей написано

скопировано 1474560 байт (1,5 MB), 0,0438156 секунд, 33,7 MB/s

Пример 2: резервирование MBR:

Команда, которая скопирует нам MBR в файл backup.mbr выглядит так:

# dd if=/dev/hda of=backup.mbr bs=512 count=1

1+0 записей считано

1+0 записей написано

скопировано 512 байт (512 B), 0,000358146 секунд, 1,4 MB/s

Резервная копия создана, теперь главное не забыть, где она лежит. И если какой-нибудь вредный дистрибутив криво установит нам загрузчик Grub, то мы спокойно проделаем обратную операцию:

dd if=backup.mbr of=/dev/hda bs=512 count=1

Пример создания файла размером 1 Гб заполненного нулям:

dd if=/dev/zero of=/mnt/disk/image.img bs=1M count=1024

Пример создания файла размером 1 Гб заполненного нулям и единицами в рандомном порядке:

dd if=/dev/urandom of=/mnt/disk/image.img bs=1M count=1024

Пример копирования дискового тома /dev/hda1 в файл образ /mnt/disk/hda1\_image.img для резерва:

dd if=/dev/hda1 of=/mnt/disk/hda1\_image.img bs=1M

Пример копирования дискового тома /dev/hda1 на дисковый том /dev/had2:

dd if=/dev/hda1 of=/dev/hda2 bs=1M

**Задания на лабораторную работу**

**Задание 1.**

**Создание виртуального жесткого диска**

1. Создать новый виртуальный диск для виртуальной машины.

Для этого необходимо зайти в настройки виртуальной машины и с помощью мастера создания виртуальных дисков создать динамически расширяющийся образ диска, а затем подключить его для использования вашей виртуальной машиной. Если диск создан и подключен, то после загрузки ОС Linux вы сможете проверить определился ли новый жесткий диск, с помощью команды:

fdisk –l

Данная команда покажет все блочные устройства подключенные к вашей виртуальной машине.

В выводе данной команды можно видеть, что нет правильной таблице разделов на диске hdb.

**Задание 2.**

**Создание таблицы разделов MBR и дискового тома**

1. Создать таблицу разделов на новом жестком диске с помощью утилиты fdisk:

Для чего выполнить команду:

fdisk /dev/hdb

Результатом выполнения команды станет вывод различной информации о состоянии диска и предложение ввести код команды для выполнения какого-либо действия, для подсказки используйте команду **m**.

Для создания пустой таблицы разделов MBR выбираем опцию **o**.

**Внимание: для того чтобы изменения были записаны на диск надо обязательно выполнить команду w.**

Когда создание таблицы разделов завершиться утилита fdisk завершить свою работу.

2. Создание раздела на HDD. Для того чтобы создать раздел опять воспользуемся утилитой fdisk, в том же порядке что для создание таблицы разделов. Только теперь нам понадобиться ее опция **n, которая позволяет создать новые разделы**. В ходе создания будут задаваться стандартные вопросы о типе раздела, его номере и конечно же размере. Размер в программе fdisk указывается в цилиндрах, если мы согласимся с использованием начального и конечного цилиндра по умолчанию, то создаваемый раздел займет весь hdd.

Результат выполнения операции можно будет просмотреть командой

fdisk –l

**Задание 3.**

**Форматирования тома и его монтирование**

**1.** Отформатировать созданный дисковый раздел в файловую систему ext3 или reiserfs c помощью команды:

mkfs.ext3 /dev/hdb1

mkreiserfs /dev/hdb1

2. Создать для него точку монтирования, например каталог **/mnt/disk1:**

mkdir /mnt/disk1

3. Примонтировать диск командой **mount:**

mount –t ext3 /dev/hdb1 /mnt/disk1

или

mount –t reiserfs /dev/hdb1 /mnt/disk1

4. Убедиться в том, что диск примонтирован командой **df –h**, которая выведет на экран все смонтированные диски и виртуальные файловые системы, с указанием их размера, а также место занятого на них.

**Задание 4.**

**Изменение размера swap**

Утилита top позволяет отслеживать состояние системы в консоле, по сути это монитор ресурсов в консольном варианте, показывающий:

* загрузку CPU
* RAM
* Файла подкачки swap
* Подробные данные о расходуемых ресурсах каждым системным процессом

Перед выполнением данного задания надо в консоли запустить команду top и проверить объем файла подкачки, после выполненного задания нужно снова будет его проверить, размер должен будет увеличиться на 1 Гб. Обязательно отобразить соответствующие скриншоты в отчете.

1. Создать с помощью утилиты dd файл размером 1024 Мегабайта, на диске, который только был создан и примонтирован. Создаваемый файл заполнить нулями.

dd if=/dev/zero of=/mnt/disk1/file1.img bs=1M count=1024

2. Отформатировать полученный файл в файловую систему swap

mkswap /mnt/disk1/file1.img

3. Подключить данный файл как swap к уже имеющемуся файлу подкачки

swapon /mnt/disk1/file.img

**Задание 5.**

**Создание пустого файла и монтирование его как файловую систему.**

1. Создать с помощью утилиты dd файл размером 1024 Мегабайта, на диске, который только был создан и примонтирован. Создаваемый файл заполнить нулями.

dd if=/dev/zero of=/mnt/disk1/file2.img bs=1M count=1024

2. Отформатировать полученный файл в файловую систему ext3

mkfs.ext3 /mnt/disk1/file1.img

3. Создать для него точку монтирования, например каталог **/mnt/disk2:**

mkdir /mnt/disk2

4. Примонтировать диск командой **mount:**

mount –t ext3 –o loop /mnt/disk1/file2.img /mnt/disk2

Стоит обратить внимание на опцию loop, дело в том что файл образ это не блочное устройство и при попытке примонтировать его без данной опции, результатом будет ошибка сообщающая о невозможности монтирования неблочных устройств без опции loop.

Такие примонтированные устройства часто называют loopback устройствами.

5. Проверить примонтировался ли диск командой

df –h

Контрольные вопросы:

1. Как с помощью утилиты dd можно зарезервировать таблицу разделов диска MBR и в случае сбоя восстановить таблицу разделов из резервной копии?
2. Как с помощью утилиты dd можно заполнить диск до состояния “ Свободно 0 байт” ?
3. Есть диск с файловой системой ext2 надо его преобразовать в файловую систему ext3 как это сделать без потери данных на диске?
4. С помощью каких утилит (консольных или графических), работающих в Linux можно изменить текущий размер диска без его форматирования и переразбития диска на новые разделы?
5. Вы создаете виртуальный диск размером 100 Гб, перед вам встал выбор чем заполнить создаваемый образ – рандомно нулями и единицами или просто нулями. Выбрать вариант при котором заполнение образа произойдет максимально быстро, обоснуйте ответ.