


## Глава 7. Работа устройствами хранения.

### 7.1. Таблица разделов.

  
учебный центр

**Таблица разделов**

- В Linux можно использовать несколько вариантов разбиения диска на разделы
  - MBR (Master Boot Record) или DOS
  - GPT (GUID Partition Table)
  - BSD/Sun
  - IRIX/SGI
- Команда `fdisk -l` показывает размер и разбиение диска на разделы

Большинство сценариев использования диска в Linux предполагает создание разделов.

Разделы на дисках создаются для логического разделения информации.

Linux может работать с диском и напрямую без создания разделов. Например при использовании в LVM (Logical Volume Manager).

В Linux в основном используются две схемы разбиения диска на разделы:

MBR (Master Boot Record) – используется для загрузки компьютера и описания разделов на диске.

GPT (GUID Partition Table) – новая схема разбиения дисков. Позволяет создать больше разделов на больших дисках.

Linux может создавать и использовать и другие типы разбиения диска, но они на практике почти не применяются: SGI и Sun.

Количество основных разделов в MBR от 1 до 4.

Обычно каждый раздел предназначен для размещения одной операционной системы, отдельной файловой системы, или, например, области размещения страниц подкачки.

MBR находится в нулевом секторе диска, а таблица разделов в нем указывает на начало, конец, размер и тип каждого раздела.

Тип раздела позволяет ОС, в которую он подключен, определить как работать с разделом. Или, другими словами, автоматизировать операции по подключению разделов на этапе загрузки.

## Глава 7. Работа устройствами хранения.

Если вы ссылаетесь на разделы в каких-нибудь командах или утилитах, то тип раздела, как правило, игнорируется.

*Примечание:* Например если вы подключаете диск с «неизвестным» разделом в Windows, то вы даже удалить этот раздел не сможете стандартными средствами Windows. Linux при работе с разделами на тип реагирует «либерально», т. е. Вы можете вручную его всегда подключить или удалить. Но если тип будет не правильный, то во время загрузки нужный вам раздел будет проигнорирован. Например раздел, который не имеет тип `fd` не будет рассматриваться как часть RAID массива.

Для обеспечения возможности создания большого количества разных файловых систем на жестком диске Linux (как и Windows) поддерживает концепцию логических разделов.

Один из основных разделов (`primary partition`) может быть объявлен, как расширенный (`extended`). В расширенном разделе может быть создано неограниченное количество логических разделов (`logical partitions`).

Разбиение диска с помощью MBR имеет существенное ограничение (2 ТБ) на размер диска, который можно разбить. Это одна из причин почему был предложен новый метод деления диска на разделы — GPT (GUID Partition Table).

GPT — часть стандарта EFI.

В GPT для обратной совместимости сохраняется MBR, в котором создается один раздел на весь диск с типом `0xee`.

После MBR в первом секторе содержится оглавление таблицы разделов. В оглавлении содержится информация о положении таблицы разделов, а также о количестве и размере записей.

Всего в GPT резервируется 128 записей.

GPT обеспечивает дублирование — оглавление и таблица разделов записаны как в начале, так и в конце диска.

Теоретически, GPT позволяет создавать разделы диска размером до 9,4 ЗБ ( $9,4 \times 10^{21}$  байт), в то время как MBR может работать только до 2,2 ТБ ( $2,2 \times 10^{12}$  байт).

Для получения таблицы разделов на диске можно воспользоваться командой `fdisk -l`.

## 7.2. Создание разделов с использованием **fdisk**.



### Создание разделов с использованием **fdisk**

- Разбиением диска на разделы можно управлять различными утилитами помимо **fdisk** это
  - **sfdisk**
  - **parted**
  - **cfdisk**
  - **gparted**
  - **blivet-gui**
- **fdisk** основной инструмент для интерактивного управления дисками

В Linux вы можете манипулировать разделами с помощью разнообразных утилит. Среди них основные:

1. **fdisk**
2. **sfdisk**
3. **cfdisk**
4. **parted/gparted**
5. **blivet-gui**

Интерактивная утилита **fdisk** позволяет оперировать с дисковыми разделами жестких магнитных дисков и обладает специальным набором собственных команд.

Утилита **sfdisk**, в отличие от **fdisk**, является утилитой для не интерактивного редактирования таблицы разделов на жестком диске.

*Примечание:* При неосторожном ее использовании легко можно утратить все данные на жестком диске, так как данные для редактирования таблицы разделов задаются в командной строке **sfdisk**.

Популярна также утилита **cfdisk**, которая позволяет редактировать таблицу разделов диска с помощью простого меню-образного интерфейса.

Утилита **parted** помимо создания и удаления разделов может перемещать разделы по диску. Может работать как интерактивно, так и не интерактивно. **gparted** — графическая утилита.

**blivet-gui** относительно новая графическая утилита для работы с разделами и файловыми системами.

Команда **fdisk** доступна только администратору. Для редактирования таблицы

## Глава 7. Работа устройствами хранения.

разделов на диске эту команду следует запустить в интерактивном режиме, указав файл устройства для требуемого жесткого диска в качестве аргумента команды:

### **Пример:**

```
# fdisk /dev/sda
```

```
Welcome to fdisk (util-linux 2.32.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.
```

```
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x806fa0ce.
```

```
Command (m for help): m
```

```
Help:
```

```
DOS (MBR)
```

- a toggle a bootable flag
- b edit nested BSD disklabel
- c toggle the dos compatibility flag

```
Generic
```

- d delete a partition
- F list free unpartitioned space
- l list known partition types
- n add a new partition
- p print the partition table
- t change a partition type
- v verify the partition table
- i print information about a partition

```
Misc
```

- m print this menu
- u change display/entry units
- x extra functionality (experts only)

```
Script
```

- I load disk layout from sfdisk script file
- O dump disk layout to sfdisk script file

```
Save & Exit
```

- w write table to disk and exit
- q quit without saving changes

```
Create a new label
```

- g create a new empty GPT partition table
- G create a new empty SGI (IRIX) partition table
- o create a new empty DOS partition table
- s create a new empty Sun partition table

***Примечание:** В этом примере команда fdisk была выполнена с аргументом - файлом устройства первого SCSI диска. Далее была выполнена встроенная команда m, отобразившая список встроенных команд fdisk.*

### Порядок работы с fdisk

- Если диск новый, то определите тип таблицы разделов o - MBR, g - GPT
- Выводится список существующих разделов на диске - p
- Удаляются ненужные разделы – d
- Создается новый раздел – n
- Если новый раздел должен иметь тип, отличный от принятого по умолчанию (83 – Linux Native), то необходимо указать тип раздела – t
- Сохранить изменения – команда w или выйти без сохранения q.

Список основных команды утилиты fdisk:

6. q - завершение работы без сохранения изменений;
7. l - вывод списка возможных типов разделов;
8. g - преобразование в GPT диск;
9. d - удаление раздела (для удаления будет запрошен номер удаляемого раздела);
10. n - создание нового раздела;
11. t - установка типа вновь созданного раздела (для установки типа необходимо ввести номер типа раздела);
12. a - выбор активного раздела;
13. w – запись измененной таблицы разделов.

Встроенная команда p утилиты fdisk выводит информацию о таблице разделов на диске.

#### **Пример:**

```
Command (m for help): p
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x806fa0ce
```

Обычная последовательность работы с утилитой fdisk для создания нового раздела на жестком диске:

## Глава 7. Работа устройствами хранения.

Выводится список существующих разделов на диске - команда `p`.

Удаляются ненужные разделы (при этом данные на них теряются безвозвратно) – команда `d`.

Создается новый раздел – команда `n`. При этом будет запрошены: тип раздела (`p` – первичный, `e` – расширенный, `l` – логический), номер раздела, первый и последний цилиндры раздела; при этом последний цилиндр нового раздела можно указать абсолютно или, после знака `+`, размер раздела в килобайтах (например, `+15000k`) или мегабайтах (`+1200M`).

Если новый раздел должен иметь тип, отличный от принятого по умолчанию (`83` – Linux Native), то необходимо указать тип раздела – команда `t` (получение списка возможных типов разделов – команда `L`).

Когда все требуемые разделы созданы, то необходимо сохранить изменения – команда `w`.

### Пример:

```
Command (m for help): p
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x806fa0ce

Command (m for help): n
Partition type
   p   primary (0 primary, 0 extended, 4 free)
   e   extended (container for logical partitions)
Select (default p): p
Partition number (1-4, default 1): 1
First sector (2048-41943039, default 2048):
Last sector, +sectors or +size{K,M,G,T,P} (2048-41943039, default 41943039): +5G
```

**Created a new partition 1 of type 'Linux' and of size 5 GiB.**

```
Command (m for help): p
Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x806fa0ce
```

Device	Boot	Start	End	Sectors	Size	Id	Type
<b>/dev/sda1</b>		<b>2048</b>	<b>10487807</b>	<b>10485760</b>	<b>5G</b>	<b>83</b>	<b>Linux</b>

```
Command (m for help): t
Selected partition 1
Hex code (type L to list all codes): L
```

0	Empty	24	NEC DOS	81	Minix / old Lin	bf	Solaris
1	FAT12	27	Hidden NTFS Win	<b>82</b>	<b>Linux swap / So</b>	c1	DRDOS/sec (FAT-
2	XENIX root	39	Plan 9	83	Linux	c4	DRDOS/sec (FAT-
3	XENIX usr	3c	PartitionMagic	84	OS/2 hidden or	c6	DRDOS/sec (FAT-
4	FAT16 <32M	40	Venix 80286	85	Linux extended	c7	Syrinx

## Глава 7. Работа устройствами хранения.

5	Extended	41	PPC PreP Boot	86	NTFS volume set da	Non-FS data
6	FAT16	42	SFS	87	NTFS volume set db	CP/M / CTOS / .
7	HPFS/NTFS/exFAT	4d	QNX4.x	88	Linux plaintext de	Dell Utility
8	AIX	4e	QNX4.x 2nd part	8e	Linux LVM	df BootIt
9	AIX bootable	4f	QNX4.x 3rd part	93	Amoeba	e1 DOS access
a	OS/2 Boot Manag	50	OnTrack DM	94	Amoeba BBT	e3 DOS R/O
b	W95 FAT32	51	OnTrack DM6 Aux	9f	BSD/OS	e4 SpeedStor
c	W95 FAT32 (LBA)	52	CP/M	a0	IBM Thinkpad hi	ea Rufus alignment
e	W95 FAT16 (LBA)	53	OnTrack DM6 Aux	a5	FreeBSD	eb BeOS fs
f	W95 Ext'd (LBA)	54	OnTrackDM6	a6	OpenBSD	ee GPT
10	OPUS	55	EZ-Drive	a7	NeXTSTEP	ef EFI (FAT-12/16/
11	Hidden FAT12	56	Golden Bow	a8	Darwin UFS	f0 Linux/PA-RISC b
12	Compaq diagnost	5c	Priam Edisk	a9	NetBSD	f1 SpeedStor
14	Hidden FAT16 <3	61	SpeedStor	ab	Darwin boot	f4 SpeedStor
16	Hidden FAT16	63	GNU HURD or Sys	af	HFS / HFS+	f2 DOS secondary
17	Hidden HPFS/NTF	64	Novell Netware	b7	BSDI fs	fb VMware VMFS
18	AST SmartSleep	65	Novell Netware	b8	BSDI swap	fc VMware VMKCORE
1b	Hidden W95 FAT3	70	DiskSecure Mult	bb	Boot Wizard hid	fd Linux raid auto
1c	Hidden W95 FAT3	75	PC/IX	bc	Acronis FAT32 L	fe LANstep
1e	Hidden W95 FAT1	80	Old Minix	be	Solaris boot	ff BBT

Hex code (type L to list all codes): **82**

**Changed type of partition 'Linux' to 'Linux swap / Solaris'.**

Command (m for help): **p**

Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disklabel type: dos

Disk identifier: 0x806fa0ce

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sda1		2048	10487807	10485760	5G	<b>82</b>	<b>Linux swap / Solaris</b>

Command (m for help): **w**

The partition table has been altered.

Calling ioctl() to re-read partition table.

[76896.597480] sda: sda1

Syncing disks.

При изменении типа таблицы разделов все существующие разделы удаляются.

Для создания GPT разделов нужно явно сменить тип разбиения командой **g**.

**Пример:** изменение таблицы разделов на GPT и создание нового GPT раздела.

**# fdisk /dev/sda**

Welcome to fdisk (util-linux 2.32.1).

Changes will remain in memory only, until you decide to write them.

Be careful before using the write command.

[77210.824171] sda: sda1

Command (m for help): **p**

Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

**Disklabel type: dos**

## Глава 7. Работа устройствами хранения.

Disk identifier: 0x806fa0ce

Device	Boot	Start	End	Sectors	Size	Id	Type
/dev/sda1		2048	10487807	10485760	5G	82	Linux swap / Solaris

Command (m for help): **g**

**Created a new GPT disklabel** (GUID: 419B9A4F-B2BD-9D47-A19F-7466E0249CC5).  
The old dos signature will be removed by a write command.

Command (m for help): **p**

Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

**Disklabel type: gpt**

Disk identifier: 419B9A4F-B2BD-9D47-A19F-7466E0249CC5

Command (m for help): **n**

Partition number (1-128, default 1): **1**

First sector (2048-41943006, default 2048):

Last sector, +sectors or +size{K,M,G,T,P} (2048-41943006, default 41943006): **+5G**

**Created a new partition 1 of type 'Linux filesystem' and of size 5 GiB.**

Command (m for help): **p**

Disk /dev/sda: 20 GiB, 21474836480 bytes, 41943040 sectors

Units: sectors of 1 \* 512 = 512 bytes

Sector size (logical/physical): 512 bytes / 512 bytes

I/O size (minimum/optimal): 512 bytes / 512 bytes

Disklabel type: gpt

Disk identifier: 419B9A4F-B2BD-9D47-A19F-7466E0249CC5

Device	Start	End	Sectors	Size	Type
/dev/sda1	2048	10487807	10485760	5G	Linux filesystem

Command (m for help): **t**

Selected partition 1

Partition type (type L to list all types): **L**

1	EFI System	C12A7328-F81F-11D2-BA4B-00A0C93EC93B
2	MBR partition scheme	024DEE41-33E7-11D3-9D69-0008C781F39F
3	Intel Fast Flash	D3BFE2DE-3DAF-11DF-BA40-E3A556D89593
4	BIOS boot	21686148-6449-6E6F-744E-656564454649
5	Sony boot partition	F4019732-066E-4E12-8273-346C5641494F
6	Lenovo boot partition	BFBFAFE7-A34F-448A-9A5B-6213EB736C22
7	PowerPC PReP boot	9E1A2D38-C612-4316-AA26-8B49521E5A8B
8	ONIE boot	7412F7D5-A156-4B13-81DC-867174929325
9	ONIE config	D4E6E2CD-4469-46F3-B5CB-1BFF57AFC149
10	Microsoft reserved	E3C9E316-0B5C-4DB8-817D-F92DF00215AE
11	Microsoft basic data	EBD0A0A2-B9E5-4433-87C0-68B6B72699C7
12	Microsoft LDM metadata	5808C8AA-7E8F-42E0-85D2-E1E90434CFB3
13	Microsoft LDM data	AF9B60A0-1431-4F62-BC68-3311714A69AD
14	Windows recovery environment	DE94BBA4-06D1-4D40-A16A-BFD50179D6AC
15	IBM General Parallel Fs	37AFFC90-EF7D-4E96-91C3-2D7AE055B174
16	Microsoft Storage Spaces	E75CAF8F-F680-4CEE-AFA3-B001E56EFC2D
17	HP-UX data	75894C1E-3AEB-11D3-B7C1-7B03A0000000
18	HP-UX service	E2A1E728-32E3-11D6-A682-7B03A0000000
19	Linux swap	0657FD6D-A4AB-43C4-84E5-0933C84B4F4F
20	Linux filesystem	0FC63DAF-8483-4772-8E79-3D69D8477DE4



## Глава 7. Работа устройствами хранения.

21 Linux server data	3B8F8425-20E0-4F3B-907F-1A25A76F98E8
22 Linux root (x86)	44479540-F297-41B2-9AF7-D131D5F0458A
23 Linux root (ARM)	69DAD710-2CE4-4E3C-B16C-21A1D49ABED3
24 Linux root (x86-64)	4F68BCE3-E8CD-4DB1-96E7-FBCAF984B709
25 Linux root (ARM-64)	B921B045-1DF0-41C3-AF44-4C6F280D3FAE
26 Linux root (IA-64)	993D8D3D-F80E-4225-855A-9DAF8ED7EA97
27 Linux reserved	8DA63339-0007-60C0-C436-083AC8230908
28 Linux home	933AC7E1-2EB4-4F13-B844-0E14E2AEF915
29 Linux RAID	A19D880F-05FC-4D3B-A006-743F0F84911E
30 Linux extended boot	BC13C2FF-59E6-4262-A352-B275FD6F7172
31 Linux LVM	E6D6D379-F507-44C2-A23C-238F2A3DF928
32 FreeBSD data	516E7CB4-6ECF-11D6-8FF8-00022D09712B
33 FreeBSD boot	83BD6B9D-7F41-11DC-BE0B-001560B84F0F
34 FreeBSD swap	516E7CB5-6ECF-11D6-8FF8-00022D09712B
35 FreeBSD UFS	516E7CB6-6ECF-11D6-8FF8-00022D09712B
36 FreeBSD ZFS	516E7CBA-6ECF-11D6-8FF8-00022D09712B
37 FreeBSD Vinum	516E7CB8-6ECF-11D6-8FF8-00022D09712B
38 Apple HFS/HFS+	48465300-0000-11AA-AA11-00306543ECAC
39 Apple UFS	55465300-0000-11AA-AA11-00306543ECAC
40 Apple RAID	52414944-0000-11AA-AA11-00306543ECAC
41 Apple RAID offline	52414944-5F4F-11AA-AA11-00306543ECAC
42 Apple boot	426F6F74-0000-11AA-AA11-00306543ECAC
43 Apple label	4C616265-6C00-11AA-AA11-00306543ECAC
44 Apple TV recovery	5265636F-7665-11AA-AA11-00306543ECAC
45 Apple Core storage	53746F72-6167-11AA-AA11-00306543ECAC
46 Solaris boot	6A82CB45-1DD2-11B2-99A6-080020736631
47 Solaris root	6A85CF4D-1DD2-11B2-99A6-080020736631
48 Solaris /usr & Apple ZFS	6A898CC3-1DD2-11B2-99A6-080020736631
49 Solaris swap	6A87C46F-1DD2-11B2-99A6-080020736631
50 Solaris backup	6A8B642B-1DD2-11B2-99A6-080020736631
51 Solaris /var	6A8EF2E9-1DD2-11B2-99A6-080020736631
52 Solaris /home	6A90BA39-1DD2-11B2-99A6-080020736631
53 Solaris alternate sector	6A9283A5-1DD2-11B2-99A6-080020736631
54 Solaris reserved 1	6A945A3B-1DD2-11B2-99A6-080020736631
55 Solaris reserved 2	6A9630D1-1DD2-11B2-99A6-080020736631
56 Solaris reserved 3	6A980767-1DD2-11B2-99A6-080020736631
57 Solaris reserved 4	6A96237F-1DD2-11B2-99A6-080020736631
58 Solaris reserved 5	6A8D2AC7-1DD2-11B2-99A6-080020736631
59 NetBSD swap	49F48D32-B10E-11DC-B99B-0019D1879648
60 NetBSD FFS	49F48D5A-B10E-11DC-B99B-0019D1879648
61 NetBSD LFS	49F48D82-B10E-11DC-B99B-0019D1879648
62 NetBSD concatenated	2DB519C4-B10E-11DC-B99B-0019D1879648
63 NetBSD encrypted	2DB519EC-B10E-11DC-B99B-0019D1879648
64 NetBSD RAID	49F48DAA-B10E-11DC-B99B-0019D1879648
65 ChromeOS kernel	FE3A2A5D-4F32-41A7-B725-ACCC3285A309
66 ChromeOS root fs	3CB8E202-3B7E-47DD-8A3C-7FF2A13CFCEC
67 ChromeOS reserved	2E0A753D-9E48-43B0-8337-B15192CB1B5E
68 MidnightBSD data	85D5E45A-237C-11E1-B4B3-E89A8F7FC3A7
69 MidnightBSD boot	85D5E45E-237C-11E1-B4B3-E89A8F7FC3A7
70 MidnightBSD swap	85D5E45B-237C-11E1-B4B3-E89A8F7FC3A7
71 MidnightBSD UFS	0394EF8B-237E-11E1-B4B3-E89A8F7FC3A7
72 MidnightBSD ZFS	85D5E45D-237C-11E1-B4B3-E89A8F7FC3A7
73 MidnightBSD Vinum	85D5E45C-237C-11E1-B4B3-E89A8F7FC3A7
74 Ceph Journal	45B0969E-9B03-4F30-B4C6-B4B80CEFF106
75 Ceph Encrypted Journal	45B0969E-9B03-4F30-B4C6-5EC00CEFF106
76 Ceph OSD	4FBD7E29-9D25-41B8-AFD0-062C0CEFF05D
77 Ceph crypt OSD	4FBD7E29-9D25-41B8-AFD0-5EC00CEFF05D
78 Ceph disk in creation	89C57F98-2FE5-4DC0-89C1-F3AD0CEFF2BE
79 Ceph crypt disk in creation	89C57F98-2FE5-4DC0-89C1-5EC00CEFF2BE

## Глава 7. Работа устройствами хранения.

80 VMware VMFS	AA31E02A-400F-11DB-9590-000C2911D1B8
81 VMware Diagnostic	9D275380-40AD-11DB-BF97-000C2911D1B8
82 VMware Virtual SAN	381CFCCC-7288-11E0-92EE-000C2911D0B2
83 VMware Virsto	77719A0C-A4A0-11E3-A47E-000C29745A24
84 VMware Reserved	9198EFFC-31C0-11DB-8F78-000C2911D1B8
85 OpenBSD data	824CC7A0-36A8-11E3-890A-952519AD3F61
86 QNX6 file system	CEF5A9AD-73BC-4601-89F3-CDEEEEE321A1
87 Plan 9 partition	C91818F9-8025-47AF-89D2-F030D7000C2C

Partition type (type L to list all types): **20**

**Changed type of partition 'Linux filesystem' to 'Linux filesystem'.**

Command (m for help): **w**


The partition table has been altered.

Calling ioctl() to re-read partition table.

[77277.892973] sda: sda1

Syncing disks.

### 7.3. Создание файловой системы.



**Создание файловой системы**

- В Linux поддерживается несколько видов ФС
  - ext2/ext3/ext4
  - XFS
  - BTRFS
  - vfat
  - msdos (fat)

Для обеспечения возможности хранения информации в виде файлов в разделе должна быть создана файловая система, то есть должно быть произведено форматирование раздела.

При форматировании формируются суперблок, массив индексных дескрипторов и выделяется пространство для блоков данных.

Изначально в GNU/Linux основной файловой системой являлась EXT2. В настоящий момент помимо EXT2 широко используются следующие файловые системы:

1. EXT3 - осовремененная версия EXT2 с поддержкой журналирования и с улучшенными показателями времени восстановления после сбоя, продвигаемая Red Hat;
2. EXT4 - следующая ступень развития файловых систем EXT;
3. XFS - высокопроизводительная файловая система для больших и очень больших объемов хранимой информации, разработанная в Silicon Graphics, распространяемая как Open Source (начиная с ядра 2.4.24 ее поддержка имеется в ядре Linux без необходимости установки специального патча);
4. BTRFS - современная файловая система, обеспечивающая такие возможности как отказоустойчивость на уровне ФС, снимки, оптимизация для дисков SSD, динамическое выделение inode.
5. JFS - высокопроизводительная файловая система от IBM, используемая при необходимости хранения очень больших объемов информации.

Чтобы подключить и использовать определенную файловую систему нужна поддержка этой ФС в ядре.

Для создания ФС поддержка со стороны ядра не обязательна, но необходимо иметь соответствующие утилиты для обслуживания этой ФС.

## Глава 7. Работа устройствами хранения.

*Примечание: если у вас есть подходящие утилиты вы можете создать на диске какую угодно ФС, но не факт, что вы сможете ее использовать (монтировать).*

## Создание файловой системы

- Для создания файловой системы на разделе используется команда `mkfs`
- По умолчанию создается `ext4`
- Опция `-t` указывает какую ФС нужно создать (необходимо иметь пакет для работы с этой ФС)
- Другие опции определяют параметры создаваемой ФС

Для создания файловой системы используется команда `mkfs`, при вызове которой в качестве аргумента должен быть задан файл устройства, соответствующий разделу жесткого диска (могут быть использованы файлы устройств для дискет и прочих блочных устройств).

Команда `mkfs` по умолчанию создает файловую систему EXT4.

**Пример:** для создания файловой системы EXT4 на втором первичном разделе диска следует выполнить такую команду (форматирование дисков - привилегия администратора):

```
# mkfs /dev/sda2
```

При успешном выполнении команды `mkfs` на экран будет выведена информация о размере блоков и их количестве в данной файловой системе, о местоположении копий суперблока и прочее.

Для создания иной файловой системы ее тип необходимо указать после опции `-t` команды `mkfs`.

**Пример:** для создания на том же разделе диска файловой системы EXT2 надо выполнить такую команду:

```
# mkfs -t ext2 /dev/sda2
```

Вместо опции `-t` для форматирования файловых систем можно использовать специализированные утилиты:

1. `mke2fs` - для создания EXT2, EXT3 и EXT4 файловых систем;
2. `mkntfs` - для создания NTFS;
3. `mkdosfs` - для форматирования под FAT.

Альтернативный способ создания разных файловых систем – использование команд `mkfs.*`:

## Глава 7. Работа устройствами хранения.

1. /sbin/mkfs.ext2
2. /sbin/mkfs.ext3
3. /sbin/mkfs.ext4
4. /sbin/mkfs.xfs

Примечание: Эти команды, в свою очередь, являются символьными ссылками, либо же жесткими связями со специализированными утилитами, упомянутыми выше.

### **Пример:**

```
# ls -F /sbin/mkfs.*
/sbin/mkfs.btrfs*   /sbin/mkfs.ext4*   /sbin/mkfs.minix*   /sbin/mkfs.vfat@
/sbin/mkfs.cramfs* /sbin/mkfs.fat*    /sbin/mkfs.msdos@   /sbin/mkfs.xfs*
/sbin/mkfs.exfat@  /sbin/mkfs.gfs2*   /sbin/mkfs.ntfs@
/sbin/mkfs.ext2*   /sbin/mkfs.hfsplus* /sbin/mkfs.reiserfs@
/sbin/mkfs.ext3*   /sbin/mkfs.jffs2*   /sbin/mkfs.ubifs*
```

Команда `mkfs` предоставляет возможность использовать опцию `-c`, которая заставляет команду перед созданием файловой системы проверять поверхность диска на наличие плохих блоков.

## 7.4. Проверка целостности файловой системы.



### Проверка целостности файловой системы

- Команда `fsck` предназначена для запуска процесса проверки ФС на наличие ошибок
- С опцией `-c` проверяется также и диск командой `badblocks`

Целостность структуры и данных файловой системы может быть нарушена в результате сбоя системы.

*Примечание:* например, ставшего следствием сбоя питания или же аппаратной неисправности.

Если работа операционной системы была прервана и файловая система не была размонтирована, то в суперблоке файловой системы остается специальный флаг (`dirty` – грязный или `not clean`), который сообщает о том, что в файловой системе возможны нарушения.

Если файловая система была размонтирована правильно и сбоев не было, то говорят, что файловая система находится в состоянии `clean` – “чистая”.

Сбой файловой системы обычно сопровождается тем, что информация, находящаяся в кэше (дисковых буферах), не синхронизируется с информацией в файловой системе.

Сбои системы могут привести к следующим проблемам в файловой системе:

1. Искажению или потере информации в блоках данных
2. Появлению в системе блоков данных, которые считаются занятыми, хотя на них не указывает ни один индексный дескриптор.
3. Наличие перекрестных ссылок на блоки данных.
4. Появлению метаданных с ненулевым счетчиком ссылок, на которые не ссылаются никакие файлы.
5. Наличие противоречивых записей в каталогах и т.п.

Различают два класса возможных нарушений в файловой системе:

1. Нарушение целостности данных.
2. Нарушение целостности структуры файловой системы.

## Глава 7. Работа устройствами хранения.

Для защиты целостности данных могут быть использованы разнообразные методы резервного хранения данных (backup).

*Примечание:* Утилиты восстановления целостности файловой системы не могут гарантировать восстановление данных после сбоя. Гарантом сохранности данных является только наличие правильно выполненных резервных копий данных.

Для восстановления структуры файловых систем в GNU/Linux используется утилита `fsck`

Для проверки различных типов файловых систем используются специализированные утилиты `fsck.*` является или опция `-t`

**Внимание!** Проверка целостности файловой системы должна осуществляться лишь на размонтированной файловой системе. Невыполнение этого требования может привести к полной потере данных на файловой системе!

### Пример:

```
# ls -F /sbin/fsck.*
/sbin/fsck.btrfs*   /sbin/fsck.ext3*   /sbin/fsck.hfs@    /sbin/fsck.ntfs@
/sbin/fsck.cramfs* /sbin/fsck.ext4*   /sbin/fsck.hfsplus* /sbin/fsck.reiserfs@
/sbin/fsck.exfat@  /sbin/fsck.fat*    /sbin/fsck.minix*  /sbin/fsck.vfat@
/sbin/fsck.ext2*   /sbin/fsck.gfs2*   /sbin/fsck.msdos@   /sbin/fsck.xfs*
```

Если вызвать утилиту `fsck` без опции `-t`, указывающей тип файловой системы, то будет вызвана утилита `e2fsck`, предназначенная для проверки файловой системы EXT.

### Пример:

```
# fsck.ext3 /dev/sda1
e2fsck 1.44.6 (5-Mar-2019)
/dev/sda1: clean, 11/327680 files, 39535/1310720 blocks
```

*Примечание:* Эта команда проверит целостность файловой системы `ext3` на первом первичном разделе первого SCSI диска в системе.

Полная проверка осуществляется не всегда, а только при условии:

1. Наличие флага `dirty` в суперблоке, что бывает при сбое или выключении питания без размонтирования файловой системы.
2. Достижения максимального разрешенного количества монтирований файловой системы без проверки ее целостности.
3. Достижения максимального срока без проверки целостности файловой системы.

Если необходимо выполнить полную проверку файловой системы в отсутствии любого из приведенных выше условий, то надо использовать опцию `-f` команды `e2fsck`.

Для проверки поверхности диска перед проверкой файловой системы требуется использовать опцию `-c` команды `e2fsck`.

**Пример:** приведенная ниже команда выполнит полную проверку файловой системы, так как превышено максимально разрешенное для данной файловой системы число монтирований без проверки целостности файловой системы.

```
# e2fsck /dev/sda1
e2fsck 1.32 (09-Nov-2002)
USBDISK has been mounted 27 times without being checked, check forced.
Pass 1: Checking inodes, blocks, and sizes
```



## Глава 7. Работа устройствами хранения.

```
Pass 2: Checking directory structure
Pass 3: Checking directory connectivity
Pass 4: Checking reference counts
Pass 5: Checking group summary information
USBDISK: 3585/2442240 files (2.2% non-contiguous), 3004291/19534832 blocks
```

Команда `e2fsck` использует специальный каталог `lost+found`, находящийся в корневом (верхнем) каталоге файловой системы устройства для сохранения потерянных цепочек блоков данных (файлов, у которых нет имени).

Для проверки целостности файловой системы XFS применяется команда `xfs_repair`.

### Пример:

```
# xfs_repair /dev/sda2
Phase 1 - find and verify superblock...
Phase 2 - using internal log
        - zero log...
        - scan filesystem freespace and inode maps...
        - found root inode chunk
Phase 3 - for each AG...
        - scan and clear agi unlinked lists...
        - process known inodes and perform inode discovery...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
        - process newly discovered inodes...
Phase 4 - check for duplicate blocks...
        - setting up duplicate extent list...
        - check for inodes claiming duplicate blocks...
        - agno = 0
        - agno = 1
        - agno = 2
        - agno = 3
Phase 5 - rebuild AG headers and trees...
        - reset superblock...
Phase 6 - check inode connectivity...
        - resetting contents of realtime bitmap and summary inodes
        - traversing filesystem ...
        - traversal finished ...
        - moving disconnected inodes to lost+found ...
Phase 7 - verify and correct link counts...
done
```

## 7.5. Монтирование файловых систем.



### Монтирование файловых систем

- Монтирование — подключение ФС к общему дереву файлов и каталогов
- Выполняется командой `mount`
- Команда `mount` вызванная без аргументов показывает таблицу монтирования

В GNU/Linux все файловые системы, доступные для работы с ними, должны быть “подцеплены” к логической структуре файлов и каталогов.

Процесс “подцепления” файловой системы, существующей на дисковом или ином блочном устройстве, в общее дерево файлов и каталогов называется монтированием.

Каталог, в который произошло “подцепление” отформатированного устройства, называется точкой монтирования.

За исключением файловых систем, для которых установлены специальные настройки в, например, файле `/etc/fstab`, монтирование файловых систем производится суперпользователем.

Стандарт FHS предписывает, что точки монтирования временных файловых систем должны находиться в каталоге `/mnt`.

Временные файловые системы для сменных носителей должны в соответствии с FHS находиться в каталоге `/media`

*Примечание:* каталог `/media/cdrom` может быть точкой монтирования для CD-ROM, а `/media/usbflash` – для флоппи диска.

Команда `mount` монтирует файловую систему указанного с помощью опции `-t` типа (по умолчанию `ext2`) в каталог - точку монтирования.

Первый аргумент команды `mount` - файл блочного устройства, на котором находится монтируемая файловая система.

Второй аргумент - точка монтирования этой файловой системы.

**Пример:** для монтирования USB диска с файловой системой EXT2 следует выполнить команду:

## Глава 7. Работа устройствами хранения.

```
# mount /dev/sda1 /mnt/usbflash
```

Если команда `mount` вызвана без аргументов, то она показывает список смонтированных файловых систем, то есть имена файлов устройств и соответствующих им точек монтирования.

### Пример:

```
# mount | grep ^/dev
/dev/vda2 on / type ext4 (rw,relatime,seclabel)
/dev/vda1 on /boot type ext4 (rw,relatime,seclabel)
/dev/sda1 on /media/usbflash type ext3 (rw,relatime,seclabel)
```

Последние версии `mount` умеют сами определять тип подключаемой ФС, если нет, то при монтировании, например, CD ROM необходимо указать тип файловой системы `iso9660` :

### Пример:

```
# mount -t iso9660 /dev/cdrom /media/cdrom
```

Монтирование файловой системы подменяет индексный дескриптор каталога - точки монтирования. До монтирования индексный дескриптор соответствует каталогу, находящемуся в файловой системе, к которой монтируется устройство. После монтирования - индексный дескриптор каталога - точки монтирования принадлежит уже смонтированной файловой системе.

### Пример:

```
# ls -ldi /media/usbflash/
529793 drwxr-xr-x. 2 root root 4096 Feb 10 12:13 /media/usbflash/

# mount /dev/sda1 /media/usbflash

# ls -ldi /media/usbflash/
2 drwxr-xr-x. 3 root root 4096 Feb 10 08:47 /media/usbflash/
```

*Примечание:* Из этого примера заметно, что до монтирования временной файловой системы в каталог `/media/usbdisk`, индексный дескриптор каталога был `529793`, а после монтирования `2`. У всех корневых каталогов файловых систем на диске индексный дескриптор имеет номер `2`. У виртуальных файловых систем корневой каталог имеет номер `inode 1`.

### Пример:

```
# ls -ldi /boot /
2 dr-xr-xr-x. 18 root root 4096 Nov  9 2019 /
2 dr-xr-xr-x.  6 root root 4096 Feb  9 10:39 /boot

# mount | grep ^/dev
/dev/vda2 on / type ext4 (rw,relatime,seclabel)
/dev/vda1 on /boot type ext4 (rw,relatime,seclabel)
/dev/sda1 on /media/usbflash type ext3 (rw,relatime,seclabel)

# ls -ldi /proc
1 dr-xr-xr-x. 91 root root 0 Feb  9 10:50 /proc
```

*Примечание:* В этом примере демонстрируется то, что для каждой файловой системы `inode` корневого

## Глава 7. Работа устройствами хранения.

каталога, то есть точки монтирования - 2.

Как и операцию монтирования, размонтировать файловые системы (без специальных настроек в `/etc/fstab`) имеет право только суперпользователь.

Для размонтирования файловой системы применяется команда `umount`, которой в качестве аргумента должен быть задан единственный аргумент - либо точка монтирования, либо файл устройства.

### **Пример:**

```
# ls /media/usbflash/  
lost+found  somedocs.txt  
[root@lin00 ~]# umount /media/usbflash/  
[root@lin00 ~]# ls /media/usbflash/
```

*Примечание: В этом примере показана работа команды `umount`. После ее выполнения в каталоге - точке монтирования больше нет доступа к файлам на временной файловой системе.*

Хорошим правилом является следующее: не следует хранить какие-либо файлы в каталогах, являющихся точками монтирования временных файловых систем.

## 7.6. Работа с разделом подкачки.



### Работа с разделом подкачки

- Создание области подкачки на разделе или файле осуществляется командой `mkswap`
- Команда `swapon` — подключает, а `swapoff` — отключает область подкачки

Раздел или файл подкачки необходимы при работе GNU/Linux для обеспечения временного перемещения страниц памяти из ОЗУ в этот раздел или файл.

Такое перемещение происходит при недостатке физической памяти.

Процесс обмена страницами памяти между ОЗУ и разделом подкачки называется `swapping`, а раздел подкачки называется `swar` – разделом.

Получить информацию об использовании раздела подкачки можно с помощью команды `swapon -s`.

#### **Пример:**

```
$ /sbin/swapon -s
```

Filename	Type	Size	Used	Priority
/dev/hda5	partition	248968	0	-1

*Примечание:* Информация, полученная от команды `swapon -s`, демонстрирует следующее: имеется раздел подкачки в первом логическом разделе Primary Master IDE диска. Размер раздела – 248 Мб, из них использовано в настоящий момент – 0.

Столбец приоритет отображает порядок использования `swar` разделов. Сначала будут использованы разделы подкачки с большим номером приоритета, затем – с меньшим.

При необходимости можно добавить в систему дополнительные области подкачки. Они могут быть размещены как в разделах дисков, так и в обычных файлах.

Создавать, подключать и отключать разделы подкачки имеет право суперпользователь.

*Примечание:* Если в системе не хватает ОЗУ для выполнения каких-либо приложений, то создание раздела подкачки может быть единственным возможным методом решения этой проблемы.

Если в системе имеется несколько жестких дисков, то рекомендуется для оптимизации

## Глава 7. Работа устройствами хранения.

производительности системы разместить разделы подкачки на нескольких дисках.

Для создания файла подкачки необходимо создать файл, заполненный нулями.

**Пример:** Приведенная ниже команда создает 128 Мб файл, заполненный нулями:

```
$ dd if=/dev/zero of=swap.file bs=1k count=131072
131072+0 входных записей
131072+0 выходных записей
$ ls -l swap.file
-rw-r--r-- 1 user1 user1 134217728 Дек 19 17:43 swap.file
```

Команда `mkswap` создает в файле или разделе (указанном при помощи файла устройства) область подкачки, специальным образом размечая ее.

**Пример:**

```
$ /sbin/mkswap swap.file
Setting up swspace version 1, size = 134213 kB
```

*Примечание:* Эта команда создала в файле `swap.file` область подкачки.

Создать область подкачки, в разделе можно лишь тогда, когда тип раздела установлен Linux Swap (82 тип в команде `fdisk`).

Для создания раздела подкачки в разделе выполняется та же команда `mkswap`

**Пример:**

```
# mkswap -c /dev/sda2
```

*Примечание:* В этом примере создается раздел подкачки на втором первичном разделе первого SCSI диска. При этом используется опция `-c`, которая перед созданием области подкачки проверяет поверхность диска на наличие плохих блоков.

Подключить созданный раздел или файл подкачки можно с помощью команды `swapon`, а отключить – с помощью команды `swapoff`.

**Пример:**

```
# swapon ~user1/swap.file

# swapon -s
Filename                                Type      Size      Used      Priority
/dev/sda5                               partition 248968    0         -1
/home/user1/swap.file                   file      131064    0         -2

# swapoff ~user1/swap.file

# swapon -s
Filename                                Type      Size      Used      Priority
/dev/sda5                               partition 248968    0         -1
```

*Примечание:* В этом примере был подключен дополнительный файл подкачки с помощью команды `swapon`, а затем этот файл был отключен командой `swapoff`.

## 7.7. Файл информации о файловых системах /etc/fstab.



### Файл информации о файловых системах /etc/fstab

- Файл /etc/fstab содержит информацию о файловых системах, которые нужно смонтировать при загрузке или по требованию пользователя
- Поля /etc/fstab:
  - <device> <mount point> <type> <options> <dump> <pass>
  - В поле <device> рекомендуется указывать UUID раздела
  - UUID можно узнать командой blkid

Конфигурационный файл /etc/fstab (filesystem table) содержит информацию о файловых системах, которые нужно смонтировать при загрузке или по требованию пользователя.

Информация в файле /etc/fstab представлена в виде таблицы:

#### Пример:

```
# cat /etc/fstab

#
# /etc/fstab
# Created by anaconda on Sat Nov  9 04:18:20 2019
#
# Accessible filesystems, by reference, are maintained under '/dev/disk/'.
# See man pages fstab(5), findfs(8), mount(8) and/or blkid(8) for more info.
#
# After editing this file, run 'systemctl daemon-reload' to update systemd
# units generated from this file.
#
UUID=1a5655e7-613b-4733-ba6b-e598633402fb /                ext4
defaults                1 1
UUID=897234d2-2307-406f-990d-a9620bcb4d1f /boot            ext4
defaults                1 2
UUID=5a4f53ca-c983-4def-8535-e2541d8419bb swap              swap
defaults                0 0
```

Строки, начинающиеся с символа # являются комментариями.

Таблица, содержащаяся в файле /etc/fstab, состоит из колонок полей, назначение которых следующее:

## Глава 7. Работа устройствами хранения.

Первое поле (`fs_spec`) содержит указатель на монтируемое устройство. Здесь может быть имя файла устройства, UUID файловой системы, UUD GPT, метка GPT или метка тома в файловой системе.

Второе (`fs_file`) указывает каталог - точку монтирования

Третье (`fs_vfstype`) содержит тип файловой системы на данном носителе.

Четвертое (`fs_mntops`) содержит опции команды `mount`, которые должны быть использованы при монтировании данной файловой системы. Для разделов `swp` это поле должно содержать `sw`.

Пятое (`fs_freq`) указывает надо ли для данной файловой системы производить автоматическое резервное копирование (`backup`) командой `dump`. Если в этом поле находится 1, то резервное копирование производится, если 0 – нет.

Шестое (`fs_passno`) предназначено для порядка проверки целостности файловых систем при загрузке операционной системы. Для корневой файловой системы в этом поле должно быть установлено значение 1. Для других файловых систем, которые необходимо проверять при загрузке, следует указать 2. Если проверка не требуется, то в этом поле ставится 0.

Ниже приведена таблица, содержащая часто используемые опции команды `mount`, указываемые в поле `fs_mntops` файла `/etc/fstab`.

Опция	Назначение
<code>defaults</code>	Установки: <code>rw,suid,dev,exec,auto,nouser,asynch</code> .
<code>asynch</code>	Асинхронный режим ввода/вывода.
<code>auto</code>	Монтировать во время загрузки.
<code>noauto</code>	Не монтировать во время загрузки.
<code>exec</code>	Разрешение выполнения файлов с машинным кодом.
<code>noexec</code>	Запрет выполнения файлов с машинным кодом.
<code>suid</code>	Бит SUID устанавливать можно.
<code>nosuid</code>	Запрещена установка битов SUID.
<code>user</code>	Обычный пользователь может монтировать устройство.
<code>nouser</code>	Монтировать разрешено только суперпользователю.
<code>ro</code>	Режим только для чтения.
<code>rw</code>	Разрешено как чтение, так и запись.

Наличие записи в файле `/etc/fstab` означает, что данная файловая система может быть смонтирована без указания обоих аргументов командной строки `mount` – файла устройства и каталога – точки монтирования. Для монтирования файловой системы, указанной в `/etc/fstab`, достаточно указать либо точку монтирования, либо файл устройства.



## Глава 7. Работа устройствами хранения.

В ОС построенных на systemd для монтирования разделов на основе `/etc/fstab` генерируются специальные юниты типа `mount`.

```
# systemctl list-units -t mount
```

UNIT	LOAD	ACTIVE	SUB	DESCRIPTION
-.mount	loaded	active	mounted	Root Mount
boot.mount	loaded	active	mounted	/boot
dev-hugepages.mount	loaded	active	mounted	Huge Pages File System
dev-mqueue.mount	loaded	active	mounted	POSIX Message Queue File System
run-user-0.mount	loaded	active	mounted	/run/user/0
sys-kernel-config.mount	loaded	active	mounted	Kernel Configuration File System
sys-kernel-debug.mount	loaded	active	mounted	Kernel Debug File System

LOAD = Reflects whether the unit definition was properly loaded.

ACTIVE = The high-level unit activation state, i.e. generalization of SUB.

SUB = The low-level unit activation state, values depend on unit type.

7 loaded units listed. Pass `--all` to see loaded but inactive units, too.  
To show all installed unit files use `'systemctl list-unit-files'`.

```
# systemctl status [-].mount
```

```
● -.mount - Root Mount
   Loaded: loaded (/etc/fstab; generated)
   Active: active (mounted) since Tue 2021-02-09 10:50:18 +05; 1 day 6h ago
     Where: /
    What: /dev/vda2
      Docs: man:fstab(5)
           man:systemd-fstab-generator(8)
```

Юнит для монтирования можно создать самостоятельно или написать специальные опции монтирования в файл `/etc/fstab`, которые сформируют юнит монтирования с нужными вам параметрами. (см. `man 5 systemd.mount`).

Использование `/etc/fstab` считается предпочтительным.

**Пример:** создадим юнит, который описывает новую точку монтирования:

```
# mkfs.ext4 /dev/sda1
```

```
mke2fs 1.44.6 (5-Mar-2019)
/dev/sda1 contains a ext3 file system
   last mounted on /media/usbflash on Wed Feb 10 12:16:19 2021
Proceed anyway? (y,N) y
Creating filesystem with 1310720 4k blocks and 327680 inodes
Filesystem UUID: a13a401f-cdf5-48e3-a277-21409ad76de0
Superblock backups stored on blocks:
...
```

```
# mkdir /storage1
```

```
# vi /etc/systemd/system/storage1.mount
```

```
# cat /etc/systemd/system/storage1.mount
```

```
[Unit]
```

```
Description=Persistent Mount Point for directory /storage1
```

```
[Mount]
```

```
What=/dev/disk/by-uuid/a13a401f-cdf5-48e3-a277-21409ad76de0
```

```
Where=/storage1
```

```
Type=ext4
```

```
Options=defaults
```

## Глава 7. Работа устройствами хранения.

```
[Install]
WantedBy=sysinit.target

# systemctl start storagel.mount

# systemctl status storagel.mount
● storagel.mount - Persistent Mount Point for directory /storagel
   Loaded: loaded (/etc/systemd/system/storagel.mount; disabled; vendor preset:>
   Active: active (mounted) since Wed 2021-02-10 17:49:04 +05; 6s ago
     Where: /storagel
    What: /dev/sda1
   Tasks: 0 (limit: 12472)
  Memory: 76.0K
   CGroup: /system.slice/storagel.mount

Feb 10 17:49:04 lin00 systemd[1]: Mounting Persistent Mount Point for directory>
Feb 10 17:49:04 lin00 systemd[1]: Mounted Persistent Mount Point for directory

# systemctl enable storagel.mount
Created symlink /etc/systemd/system/sysinit.target.wants/storagel.mount →
/etc/systemd/system/storagel.mount.

# systemctl status storagel.mount
● storagel.mount - Persistent Mount Point for directory /storagel
   Loaded: loaded (/etc/systemd/system/storagel.mount; enabled; vendor preset: >
   Active: active (mounted) since Wed 2021-02-10 17:49:04 +05; 2min 17s ago
   ...
```

## 7.8. Мониторинг дисковых ресурсов.



### Мониторинг дисковых ресурсов

- `df` — использование ФС
- `du` — размер содержимого каталога
- `lsblk` - информация о блочных устройствах
- `blkid` — UUID, метки и типы ФС на разделах
- `file -s <device>` — параметры ФС

Команда `lsblk` выдает информацию о дисках, разделах и точках монтирования.

#### Пример:

```
# lsblk
NAME MAJ:MIN RM SIZE RO TYPE MOUNTPOINT
sda   8:0    0  20G  0 disk
├─sda1 8:1    0    5G  0 part /storage1
└─sda2 8:2    0    2G  0 part /storage2
sr0   11:0    1 1024M  0 rom
vda   253:0   0  20G  0 disk
├─vda1 253:1   0    1G  0 part /boot
├─vda2 253:2   0   17G  0 part /
└─vda3 253:3   0    2G  0 part [SWAP]
```

```
# lsblk -f
NAME FSTYPE LABEL UUID MOUNTPOINT
sda
├─sda1 ext4      a13a401f-cdf5-48e3-a277-21409ad76de0 /storage1
└─sda2 xfs       e5b66aa7-c101-4bcd-bb63-2d10d2e4aa44 /storage2
sr0
vda
├─vda1 ext4    BOOT  897234d2-2307-406f-990d-a9620bcb4d1f /boot
├─vda2 ext4    ROOT  1a5655e7-613b-4733-ba6b-e598633402fb /
└─vda3 swap    SWAP1 5a4f53ca-c983-4def-8535-e2541d8419bb [SWAP]
```

Команда `findmnt` показывает точки монтирования в удобном для восприятия виде.

#### Пример:

```
# findmnt
TARGET SOURCE FSTYPE OPTIONS
```

## Глава 7. Работа устройствами хранения.

/	/dev/vda2	ext4	rw,relatime,seclabel
├─/sys	sysfs	sysfs	rw,nosuid,nodev,noexec,
│   └─/sys/kernel/security	securityfs	securityfs	rw,nosuid,nodev,noexec,
│   └─/sys/fs/cgroup	tmpfs	tmpfs	ro,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/systemd	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/perf_event	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/pids	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/freezer	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/cpuset	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/net_cls,net_prio	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/cpu,cpuacct	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/rdma	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/devices	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/memory	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/blkio	cgroup	cgroup	rw,nosuid,nodev,noexec,
│       └─/sys/fs/cgroup/hugetlb	cgroup	cgroup	rw,nosuid,nodev,noexec,
└─/sys/fs/pstore	pstore	pstore	rw,nosuid,nodev,noexec,
└─/sys/fs/bpf	bpf	bpf	rw,nosuid,nodev,noexec,
└─/sys/fs/selinux	selinuxfs	selinux	rw,relatime
└─/sys/kernel/debug	debugfs	debugfs	rw,relatime,seclabel
└─/sys/kernel/config	configfs	configfs	rw,relatime
├─/proc	proc	proc	rw,nosuid,nodev,noexec,
│   └─/proc/sys/fs/binfmt_misc	systemd-1	autofs	rw,relatime,fd=35,pgrp=
├─/dev	devtmpfs	devtmpfs	rw,nosuid,seclabel,size
│   └─/dev/shm	tmpfs	tmpfs	rw,nosuid,nodev,seclabe
│   └─/dev/pts	devpts	devpts	rw,nosuid,noexec,relati
│   └─/dev/mqueue	mqueue	mqueue	rw,relatime,seclabel
│   └─/dev/hugepages	hugetlbfs	hugetlb	rw,relatime,seclabel,pa
├─/run	tmpfs	tmpfs	rw,nosuid,nodev,seclabe
│   └─/run/user/0	tmpfs	tmpfs	rw,nosuid,nodev,relatim
├─/boot	/dev/vda1	ext4	rw,relatime,seclabel
├─/storage1	/dev/sda1	ext4	rw,relatime,seclabel
└─/storage2	/dev/sda2	xfs	rw,relatime,seclabel,at

Команда `blkid` находит и печатает атрибуты блочных устройств.

### Пример:

```
# blkid
/dev/vda1: LABEL="BOOT" UUID="897234d2-2307-406f-990d-a9620bcb4d1f" TYPE="ext4"
PARTUUID="706c87eb-01"
/dev/vda2: LABEL="ROOT" UUID="1a5655e7-613b-4733-ba6b-e598633402fb" TYPE="ext4"
PARTUUID="706c87eb-02"
/dev/vda3: LABEL="SWAP1" UUID="5a4f53ca-c983-4def-8535-e2541d8419bb" TYPE="swap"
PARTUUID="706c87eb-03"
/dev/sda1: UUID="a13a401f-cdf5-48e3-a277-21409ad76de0" TYPE="ext4"
PARTUUID="87ce5033-6121-0b42-b875-05ff23508aad"
/dev/sda2: UUID="e5b66aa7-c101-4bcd-bb63-2d10d2e4aa44" TYPE="xfs"
PARTUUID="fea45c49-5143-a34b-8b92-9e78bc409109"

# blkid -i /dev/sda1
/dev/sda1: MINIMUM_IO_SIZE="512" PHYSICAL_SECTOR_SIZE="512"
LOGICAL_SECTOR_SIZE="512"

# blkid -t TYPE="ext4"
/dev/vda1: LABEL="BOOT" UUID="897234d2-2307-406f-990d-a9620bcb4d1f" TYPE="ext4"
PARTUUID="706c87eb-01"
/dev/vda2: LABEL="ROOT" UUID="1a5655e7-613b-4733-ba6b-e598633402fb" TYPE="ext4"
PARTUUID="706c87eb-02"
```

## Глава 7. Работа устройствами хранения.

```
/dev/sda1: UUID="a13a401f-cdf5-48e3-a277-21409ad76de0" TYPE="ext4"  
PARTUUID="87ce5033-6121-0b42-b875-05ff23508aad"
```

Команда `df` выводит количество свободного места в блоках на указанном устройстве, а если оно не указано, то на всех смонтированных файловых системах.

Удобно использовать опцию `-h`, для отображения информации в понятных для пользователя единицах (`human readable format`):

### **Пример:**

```
$ df -h  
Filesystem      Size  Used Avail Use% Mounted on  
devtmpfs        975M    0  975M   0% /dev  
tmpfs           989M    0  989M   0% /dev/shm  
tmpfs           989M   17M  973M   2% /run  
tmpfs           989M    0  989M   0% /sys/fs/cgroup  
/dev/vda2       17G   3.2G   13G  21% /  
/dev/vda1       976M  163M  747M  18% /boot  
tmpfs           198M    0  198M   0% /run/user/0  
/dev/sda1       4.9G   20M   4.6G   1% /storage1  
/dev/sda2       2.0G   47M   2.0G   3% /storage2
```

Для получения информации о наличии свободных индексных дескрипторов необходимо вызвать команду `df -i`:

### **Пример:**

```
$ df -i  
Filesystem      Inodes   IUsed   IFree IUse% Mounted on  
devtmpfs        249450    376  249074    1% /dev  
tmpfs           253090     1  253089    1% /dev/shm  
tmpfs           253090    513  252577    1% /run  
tmpfs           253090     17  253073    1% /sys/fs/cgroup  
/dev/vda2       1114112 138925  975187   13% /  
/dev/vda1        65536    320   65216    1% /boot  
tmpfs           253090     5  253085    1% /run/user/0  
/dev/sda1       327680    11  327669    1% /storage1  
/dev/sda2      1048576     3 1048573    1% /storage2
```

*Примечание:* Обозначения *K* и *M* присутствующие в листинге выше следует понимать, как тысячи и миллионы индексных дескрипторов.

Для того, чтобы узнать сколько пространства занимают файлы в каталоге следует использовать команду `du`, отображающую количество блоков, занимаемое каждым каталогом и каждым его подкаталогом.

Можно использовать `du -h` для отображения информации в удобных единицах:

### **Пример:**

```
$ du -h /etc/rc.d  
161K  /etc/rc.d/init.d  
1.0K  /etc/rc.d/rc0.d  
1.0K  /etc/rc.d/rc1.d  
1.0K  /etc/rc.d/rc2.d  
1.0K  /etc/rc.d/rc3.d  
1.0K  /etc/rc.d/rc4.d  
1.0K  /etc/rc.d/rc5.d  
1.0K  /etc/rc.d/rc6.d
```

## Глава 7. Работа устройствами хранения.

```
18K      /etc/rc.d/scripts
207K     /etc/rc.d
```

Также можно отобразить лишь суммарную информацию о каталоге, без вывода подробностей о подкаталогах. Для этого используется `du -s`

### **Пример:**

```
$ du -sh ~
467M    /home/user1
```

*Примечание: В этом примере получена информация о суммарном пространстве, используемом домашним каталогом пользователя.*

## 7.9. Квотирование дискового пространства пользователей.



### Квотирование дискового пространства пользователей

- Вы можете включить квоты на разделе для пользователей или групп
- Можно квотировать объем или количество файлов
- Для работы с квотами диска потребуется пакет quota и поддержка квот в ядре для данной ФС

Если в системе работает множество пользователей, то даже при большом объеме накопителей может обнаружиться недостаток дискового пространства, вызванный деятельностью пользователей.

Возможный подход к решению данной проблемы заключается в создании специального раздела для каталогов пользователей.

*Примечание:* В таком случае, естественно, файлы пользователей не смогут превзойти размер раздела. Однако такое решение создаст иную проблему – одни пользователи будут фактически лишать других пользователей дискового пространства.

Другое решение состоит во введении в системе квотирования.

Квотирование позволяет ограничить размер дискового пространства, занимаемого файлами пользователя, выделяя пользователю определенное количество дисковых ресурсов – квоту.

Для установки системы квотирования, прежде всего, необходимо проверить, установлен ли пакет quota (пакет портирован из FreeBSD).

**Пример.** В системах, использующих в качестве менеджера пакетов RPM, это можно сделать с помощью команды `rpm -q quota`. Эта команда должна вывести полное имя пакета с его версией.

```
$ rpm -q quota
quota-4.04-12.el8.x86_64
```

В простейшем случае достаточно проверить командой `which quota` нет ли исполняемого файла `quota` в каталогах, указанных в переменной окружения `PATH`.

Помимо наличия пакета `quota`, который позволяет проверять объем дисковых ресурсов, занимаемых пользователем, в системе должно быть установлено ядро с

## Глава 7. Работа устройствами хранения.

поддержкой квоты. Поддержка квоты включается в разделе конфигурирования файловых систем при настройке ядра перед его сборкой.



### Порядок применения квот

- Определение файловых систем, ресурсы которых будут квотированы, через опции монтирования
- Создание базы данных квот для каждой квотируемой файловой системы.
- Установки индивидуальных значений квот для пользователей и групп.

Процесс создания квот состоит из трех шагов:

1. Определение файловых систем, ресурсы которых будут установлены квоты.
2. Создание базы данных квот для каждой квотируемой файловой системы.
3. Установки индивидуальных значений квот для пользователей и групп.
4. Запуск квотирования в ядре.

Фактически, определение квотируемых файловых систем заключается в их монтировании с опциями:

1. `usrquota` - для установки индивидуальных пользовательских квот;
2. `grpquota` - для установки квот для групп пользователей.

Эти опции могут быть установлены вместе или по отдельности в зависимости от требований, предъявляемых к системе. Команда монтирования допускает эти опции, но игнорирует их, так как они предназначены для программного обеспечения квотирования.

Чаще всего поддержка квот включается при монтировании файловых систем, поэтому опции `usrquota` и `grpquota` указывают для квотируемых файловых систем в файле `/etc/fstab`.

#### **Пример:**

```
UUID=a13a401f-cdf5-48e3-a277-21409ad76de0 /storage1 ext4 defaults,usrquota 0 0
```

*Примечание: В этом примере для каталога `/storage1` будут установлены пользовательские квоты.*

После внесения изменений в файл `/etc/fstab` квотируемая файловая система должна быть перемонтирована для того, чтобы опции квотирования вступили в силу.

#### **Пример:**

## Глава 7. Работа устройствами хранения.

```
# mount -o remount /storage1
```

Для проверки правильности монтирования следует выполнить команду `mount` без каких-либо аргументов. В списке смонтированных файловых систем эта команда должна отобразить опции квотирования для тех файловых систем, которые подлежат квотированию.

### Пример:

```
# mount | grep storage1  
/dev/sda1 on /storage1 type ext4 (rw,relatime,seclabel,quota,usrquota)
```

После монтирования котируемых файловых систем с установленными опциями квотирования необходимо создать базу данных квот, в которой хранится информация о файловых ресурсах, занимаемых пользователями.

База данных хранится в файлах:

1. `aquota.user` - для пользовательских квот (версия 2 и более пакета `quota`);
2. `aquota.group` - для квот групп пользователей.

Файлы базы данных квот должны располагаться в каталогах - точках монтирования котируемых файловых систем.

При использовании квоты версии 2 и более файлы базы данных будут созданы автоматически с помощью команды `quotacheck`, которая предназначена для проверки целостности базы данных квот.

*Примечание:* Обычно эта команда вызывается автоматически при монтировании котируемых файловых систем во время загрузки операционной системы.

Наиболее часто используемые опции команды `quotacheck`:

1. `-c` - создать новую базу данных квот. При использовании этой опции предыдущие настройки квот для пользователей и групп будут уничтожены.
2. `-m` - не перемонтировать файловую систему в режиме только для чтения при создании или проверке квот (этой опции нет в квоте версии 1).
3. `-a` - проверить квоты на всех смонтированных с опциями квотирования файловых системах. Если эта опция не используется, то для команды должен быть аргумент, указывающий котируемую файловую систему, в которой осуществляется проверка.
4. `-u` - проверить только пользовательские квоты.
5. `-g` - проверить только квоты для групп.
6. `-v` - выдавать дополнительную информацию в процессе работы.

**Пример:** Показанная ниже команда создает новую базу данных пользовательских квот во всех котируемых файловых системах при использовании квоты версии 2 без перемонтирования этих файловых систем только для чтения:

```
# quotacheck -caumv  
quotacheck: Your kernel probably supports journaled quota but you are not using it. Consider switching to journaled quota to avoid running quotacheck after an unclean shutdown.  
quotacheck: Scanning /dev/sda1 [/storage1] done  
quotacheck: Cannot stat old user quota file /storage1/aquota.user: No such file or directory. Usage will not be subtracted.  
quotacheck: Cannot stat old group quota file /storage1/aquota.group: No such
```

## Глава 7. Работа устройствами хранения.

```
file or directory. Usage will not be subtracted.
quotacheck: Checked 3 directories and 0 files
quotacheck: Old file not found.
```

*Примечание:* Обратите внимание на выделенную строку. Поэтому поменяем опцию квотирования в `/etc/fstab`:

```
UUID=a13a401f-cdf5-48e3-a277-21409ad76de0 /storage1 ext4
defaults,usrjquota=aquota.user,jqfmt=vfsv0 0 0

# mount -o remount /storage1
[127071.443316] EXT4-fs (sda1): old and new quota format mixing
mount: /storage1: mount point not mounted or bad option.

# umount /storage1
# mount /storage1

# quotacheck -caumv
quotacheck: Scanning /dev/sda1 [/storage1] done
quotacheck: Old group file name could not been determined. Usage will not be
subtracted.
quotacheck: Checked 3 directories and 1 files
```

При использовании файловой системы `ext4` вы при выполнении команды `quotacheck -caumv` увидите следующее сообщение *«quotacheck: Your kernel probably supports ext4 quota feature but you are using external quota files. Please switch your filesystem to use ext4 quota feature as external quota files on ext4 are deprecated.»*.

Для включения квоты в `ext4` необходимо включить опцию квотирования в суперблоке:

```
# tune2fs -O quota /dev/sda1
```

Для проверки можно выполнить команду:

```
# tune2fs -l /dev/sda1 | grep features
Filesystem features:      has_journal ext_attr resize_inode dir_index filetype
needs_recovery extent 64bit flex_bg sparse_super large_file huge_file dir_nlink
extra_isize quota metadata_csum
```

В файле `/etc/fstab` опций квотирования указывать в этом случае не требуется.

Далее следует приступить к определению квот для пользователей и/или групп.

Квотированию могут подлежать:

1. суммарный объем файлов данного пользователя или группы пользователей;
2. количество `inode`, то есть количество файлов, пользователя или группы пользователей.

В пакете квотирования используется модель ограничений, в которой указывают ограничения:

1. которое не может быть превзойдено (`hard quota`)
2. при котором пользователь получает сообщение о превышении квоты, но блокирование записи еще не производится (`soft quota`).

Время, на которое разрешается превышать `soft quota`, называется `grace period`. По умолчанию он установлен равным 7 суткам.

По истечении этого срока даже если `hard quota` не достигнута операции записи

## Глава 7. Работа устройствами хранения.

блокируются.

Этот период времени не может быть установлен индивидуально для каждого пользователя - он распространяется на всех.

Таким образом, установка индивидуальных квот для пользователей или групп заключается в следующем:

1. Определяется grace period для всех пользователей и групп. Если эта операция не производится, то используется значение по умолчанию.
2. Устанавливаются настройки soft и hard квот на объем файлов и их количество (inode) для одного или нескольких пользователей индивидуально. Любой из них может быть использован в качестве образца для установки квот для других пользователей.
3. Используя настройки квот для одного из пользователей как шаблон, производится определение квот для других пользователей.

Продолжительность grace period может быть установлена с помощью команды `edquota -t`. При вызове этой команды будет запущен текстовый редактор по умолчанию (например, `vi`), в окне редактирования которого можно будет изменить текущее значение grace period для количества и объема файлов.

*Примечание:* Редактирование производится во временном файле, причем если работа редактора будет завершена с записью в этот временный файл, то команда `edquota` считает содержимое временного файла и установит новое значение grace period.

Значение периода времени grace period может быть задано в следующих единицах времени:

1. seconds - секунды;
2. minutes - минуты;
3. hours - часы;
4. days - дни;
5. weeks - недели;
6. months - месяцы.

После установки grace period можно перейти к установке пользовательских и групповых квот. Пользовательская квота настраивается с помощью команды `edquota -u`.

### **Пример:**

```
# edquota -u test4
```

*Примечание:* В данном случае файловая квота будет установлена для пользователя `test4`.

Как и при установке grace period, будет вызван редактор по умолчанию, в окне редактирования которого можно будет увидеть уже занятый файлами пользователя объем дискового пространства и количество использованных inode. Также будут отображены столбцы для установки новых значений soft и hard квот для суммарного объема файлов и количества inode. Именно эти столбцы должны быть отредактированы для установки новых значений. После выхода из редактора с сохранением временного файла он будет считан командой `edquota`, после чего будут установлены новые значения квот.

Как только квота установлена для одного из пользователей системы, настройки для

## Глава 7. Работа устройствами хранения.

него могут быть использованы в качестве шаблона для других пользователей. Для этого используется команда `edquota -p <шаблон>`.

*Примечание:* Преимуществом этой команды является неинтерактивный режим работы. То есть, при вызове этой команды редактор текста не запускается, а настройки для пользователя, используемые как шаблон, будут просто скопированы в базе данных квот для других пользователей, указанных после опции `-u`.

### **Пример:**

```
# edquota -p test4 -u test1 test2 test3
```

Если требуется установить квоты для групп пользователей, то вместо опции `-u` команды `edquota` надо использовать опцию `-g`, после которой должна быть указана группа пользователей для квотирования.

После определения пользовательских и/или групповых квот необходимо включить программное обеспечение квотирования в ядре. Это достигается с помощью вызова команды `quotaon`.

Если в командной строке не будет указана опция `-a`, включающая квотирование для всех смонтированных с опциями квотирования файловых систем, то в качестве аргумента должна быть указана котируемая файловая система.

### **Пример:**

```
# quotaon -vu /dev/sda1
```

*Примечание:* Эта команда включит квотирование файловой системы `/dev/sda1`. Опция `-u` указывает, что должны быть включены пользовательские квоты, а опция `-v` выдает сообщение о включении квоты.

Вызов команды `quotaon` обычно осуществляется автоматически при загрузке операционной системы после монтирования файловых систем с установленными опциями квотирования.

Выключить квотирование можно с помощью команды `quotaoff`. Для нее также необходимо либо указать опцию `-a` для отключения квотирования всех файловых систем, либо указать котируемую файловую систему.

Пользователь может получить информацию об ограничениях, установленных квотой для него с помощью команды `quota`.

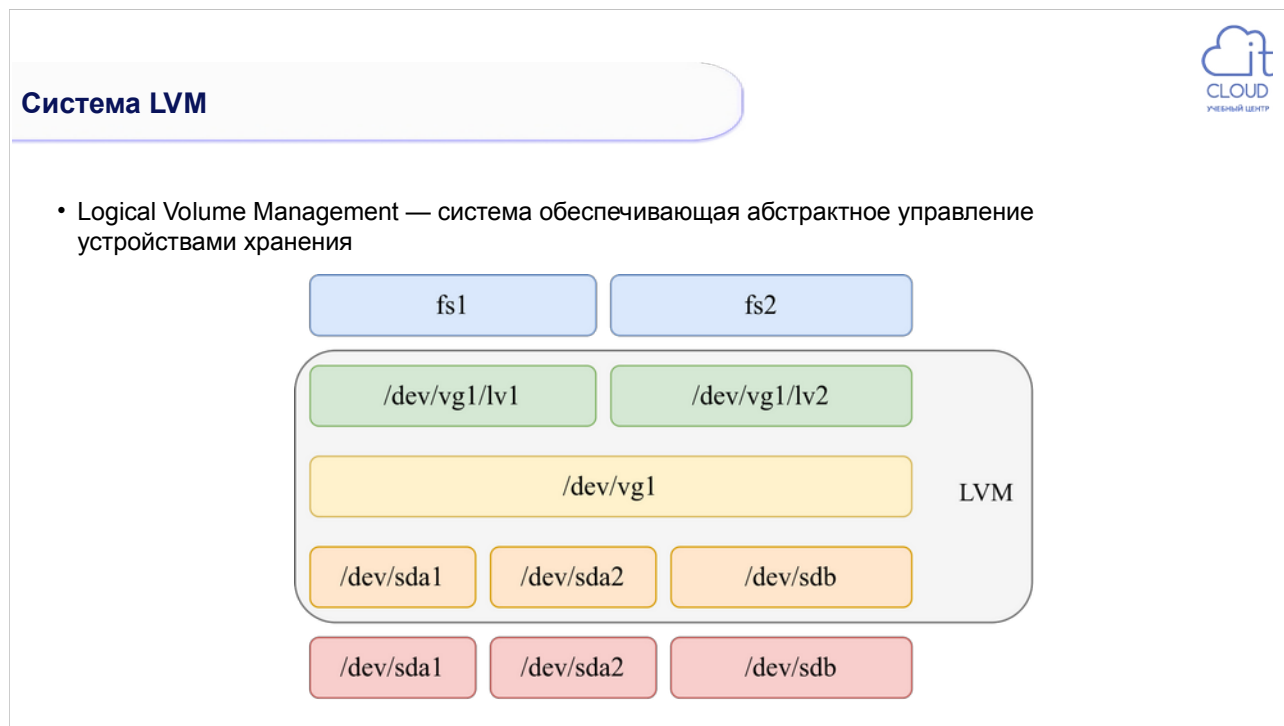
*Примечание:* Она сообщает об объеме и количестве файлов пользователя. Если пользователь превысит значение `soft quota`, то ему выдается соответствующее сообщение. С этого момента начнется обратный отсчет времени `grace period`. Как только этот период времени будет исчерпан, или же при попытке превысить `hard quota`, операции записи будут блокироваться.

Администратор может получать информацию о состоянии пользовательских квот с помощью команды `repquota`. Если для этой команды не установлена опция `-a`, выводящая информацию по всем котируемым файловым системам, то в качестве аргумента этой команды должна быть установлена котируемая файловая система.

## Глава 8. Программные системы хранения

### 8.1. Система LVM.

#### 8.1.1. Преимущества использования LVM



Система логического управления томами LVM (Logical Volume Management) обеспечивает дополнительный уровень абстракции от дисковых разделов и физических накопителей в системе. Вместо работы с системой дисковых разделов LVM предполагает операции с логическими томами, объединяющими, возможно, несколько разделов или жестких дисков.

Использование LVM позволяет объединять физические накопители в логические тома, обеспечивая более удобную работу с ними.

При использовании LVM администратор работает не с обычными файлами устройств, имена которых никоим образом не отражают суть хранимых на них данных, а с логическими устройствами, которым могут быть назначены удобные имена.

Система LVM обеспечивает удобные возможности изменения размеров логических томов, существенно упрощая таким образом планирование исходного разбиения жесткого диска на разделы при установке системы.

Система LVM обеспечивает возможность получения так называемых snapshots - моментальных снимков состояния логического тома. Суть их состоит в фиксации информации на диске в определенный момент времени. Это используется, например, при необходимости резервного копирования на лету.

### 8.1.2. Терминология LVM

Наивысший уровень абстракции в LVM - это группа томов (volume group VG). Она объединяет наборы логических и физических томов в единую административную сущность.

Физический том (physical volume - PV) - обычно жесткий диск, устройство RAID или раздел жесткого диска.

Логический том (logical volume - LV) - субъект LVM, содержащий конкретную файловую систему. Доступен, как блочный файл устройства.

Физический экстенд (physical extent - PE) - последовательность блоков жесткого диска, предназначенная для хранения одного какого-либо файла. Физические экстенды имеют одинаковые размеры со всеми логическими экстендами в одной группе томов.

Логический экстенд (logical extent - LE) - непрерывная последовательность блоков логического тома. Размер логических экстендов одинаков для всех логических томов в группе томов.

Используется несколько стратегий отображения логических экстендов в физические:

1. Линейное отображение, в случае которого логические экстенды размещаются на физических томах последовательно. То есть, например, экстенды с первого по сотый - на первом физическом томе, а со сто первого по двухсотый - на втором физическом томе.
2. Отображение с чередованием, в котором допускается части логических экстендов размещать на различных физических томах.
3. Зеркалирование, в котором данные экстендов дублируются.
4. Различные варианты RAID.
5. Снапшоты — специальный том в который записываются обратные изменения произошедшие в томе, для которого создан снапшот.
6. Тонкие тома (thin) для тома место на дисках выделяется по мере необходимости.
7. Кэш, в котором сочетаются быстрые SSD и медленные HDD устройства.

### 8.1.3. Использование LVM.



#### Система LVM

- Последовательность действий
  - 1) Создать PV — `pvcreate`
  - 2) Создать VG — `vgcreate`
  - 3) Создать LV — `lvcreate`
- Получение информации о группах, томах — `vgdisplay`, `pvdisk`, `lvdisplay`
- Управление группами и томами — `vg*`, `pv*`, `lv*`

Перед использованием LVM физический диск или дисковый раздел необходимо инициализировать командой `pvcreate`.

#### **Пример:**

```
pvcreate /dev/sdd
```

Если для LVM используется раздел диска, то вначале необходимо установить с помощью команды `fdisk` тип этого раздела `0x8e`. Далее следует использовать команду `pvcreate` аналогично тому, как она применялась при инициализации диска.

#### **Пример:**

```
pvcreate /dev/sda5
```

Далее требуется создать группу томов (VG). Для этого используется команда `vgcreate`.

#### **Пример:** Создание группы томов на двух устройствах:

```
vgcreate testvg /dev/hda2 /dev/hdb1
```

В случае использования в системе `devfs` необходимо использовать реальные имена файлов устройств вместо символических ссылок на эти файлы.

#### **Пример:** Для системы, использующей `devfs`:

```
vgcreate testvg /dev/ide/host0/bus0/target0/lun0/part2 \  
/dev/ide/host0/bus0/target1/lun0/part1
```

Для вновь созданной группы томов будет установлен размер экстенда по умолчанию, равный 4 Mb. При необходимости использования иного размера экстенда следует указать требуемый размер после ключа `-s`.

После создания группы томов необходимо активировать эту группу томов. Это позволяет сделать команда `vgchange -ay`.



### **Пример:** Активирование группы томов:

```
vgchange -ay testvg
```

Эта же команда `vgchange`, но с ключами `-an` позволяет деактивировать группу томов.

### **Пример:** Деактивация группы томов:

```
vgchange -an testvg
```

После деактивации группы ее можно удалить с помощью команды `vgremove`. Однако, перед удалением следует убедиться, что в этой группе отсутствуют логические тома.

### **Пример:** Удаление группы томов:

```
vgremove testvg
```

Команда `vgextend` позволяет добавлять физический том в существующую группу томов. Добавляемый физический том должен быть инициализирован.

### **Пример:** добавление физического тома `/dev/sdd` в группу томов `testvg`

```
vgextend testvg /dev/sdd
```

Существует возможность удаления физического тома из группы томов. Это можно сделать с помощью команды `vgreduce`.

Перед удалением физического тома из группы томов необходимо убедиться в том, что данный физический том не используется ни одним логическим томом. Такую информацию можно получить, используя команду `pvdisplay`.

### **Пример:**

*Проверка используется ли физический том:*

```
pvdisplay /dev/sdd
```

*Удаление физического тома:*

```
vgreduce testvg /dev/sdd
```

Команды `vgdisplay` и `pvdisplay` предоставляют информацию, соответственно, о состоянии группы томов и физических томов в группе. Используя полученную с помощью этих команд информацию следует решить, на каких физических томах будут созданы логические тома.

Собственно создание логического тома осуществляется командой `lvcreate`, причем логический том может быть создан с использованием как линейного отображения на физические тома, так и с использованием чередования.

### **Пример:**

```
lvcreate -L2000 -nfirstlv testvg
```

**Примечание:** данная команда создаст линейный логический том (LV), принадлежащий группе (VG) `testvg`. Логический том будет в данном случае именоваться, как `firstlv`, и ему будет соответствовать блочное устройство `/dev/testvg/firstlv`. Размер тома - 2 Гб.

```
lvcreate -l100 -i2 -I16 -nsecondlv testvg
```

**Примечание:** Команда, приведенная выше, создаст логический том с чередованием двух участков (опция `-i2`). Каждый участок имеет размер 16 Кб. Размер данного логического тома определяется количеством логических экстендов (LE), установленному в 50 (опция `-l100`).

Если необходимо удалить логический том, то для этого можно использовать команду `lvremove`.

### **Пример:**

```
umount /dev/testvg/firstlv
```

## Глава 8. Программные системы хранения

```
lvremove /dev/testvg/firstlv
```

**Примечание:** В этом случае будет удален логический том /dev/testvg/firstlv.

Размер логического тома может быть увеличен с помощью команды `lvextend`, которой следует указать после опции `-L` либо размер, на который необходимо увеличить размер логического тома (`-L+1G`), либо желаемый размер увеличиваемого тома (`-L10G`).

### **Пример:**

```
lvextend -L+1G /dev/testvg/secondlv
```

**Примечание:** Команда с использованием размера, на который увеличивается том:

```
lvextend -L10G /dev/testvg/secondlv
```

**Примечание:** Команда, увеличивающая размеры логического тома до 10 Гб:

Увеличение логического тома не означает автоматическое изменение размера файловой системы на нем.

Размер файловой системы могут быть изменены с помощью утилит, предназначенных для конкретных файловых систем: -

1. `ext` - утилита `resize2fs`;
2. `reiserfs` - утилита `resize_reiserfs`;
3. `xfs` - утилита `xfs_growfs`.

Файловая система `ext` может быть как размонтирована так и смонтирована перед изменением ее размера.

При использовании `xfs` она может быть подвергнута изменению размера только тогда, когда она смонтирована. Причем, при изменении размера `xfs` нельзя указывать файл устройства. Вместо него необходимо указать точку монтирования:

### **Пример:**

```
xfs_growfs /home
```

**Примечание:** Здесь будет произведено изменение размера файловой системы `xfs`, смонтированной в каталоге `/home`.

Пакет LVM предоставляет команду `lvreduce`, позволяющую уменьшать размеры логических томов. Размер логического тома не может быть меньше, чем размер файловой системы на нем.

ФС `xfs` уменьшать нельзя.

### **Пример:**

```
lvreduce -L-2G /dev/testvg/secondlv
```

**Примечание:** Данная команда уменьшает размер логического тома на 2 Гб.

Уменьшение размера тома производится в три этапа.

1. Уменьшается ФС на томе, так чтобы размер ФС был чуть меньше предполагаемого размера тома
2. Уменьшается сам том.
3. ФС увеличивается, чтобы занять весь том.

### **Пример:**

```
resize2fs /dev/testvg/firstlv 990M
```

**Примечание:** Только для размонтированной файловой системы `ext2`. Обратите внимание, что ФС уменьшается на немного меньший размер, чем ожидаемое уменьшение тома. Это делается для снижения

## Глава 8. Программные системы хранения

*вероятности повреждения ФС.*

```
lvreduce -L1G /dev/testvg/firstlv
```


Примечание: После уменьшения тома вновь расширим ФС на томе.

```
resize2fs /dev/testvg/firstlv
```

## 8.2. RAID массивы.

### 8.2.1. Использование RAID.

**RAID массивы**



- Могут обеспечивать отказоустойчивость или производительность
- Уровни определяют способ чтения-записи информации в массиве

Избыточные массивы независимых (недорогих) дисков (Redudant Array of Independent (Inexpensive) Disks) применяются в нескольких случаях:

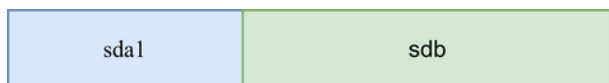
1. при необходимости обеспечения надежного хранения данных;
2. при необходимости повышения производительности дисковой подсистемы;
3. для обеспечения возможности горячей замены дисков в массиве.

Массивы RAID могут быть организованы как аппаратно с помощью специальных контроллеров, так и программно.

Варианты организации RAID массивов, предназначенные для выполнения определенного класса задач, отличающиеся организацией, называются уровнями RAID. Наиболее известны следующие уровни:

## RAID массивы

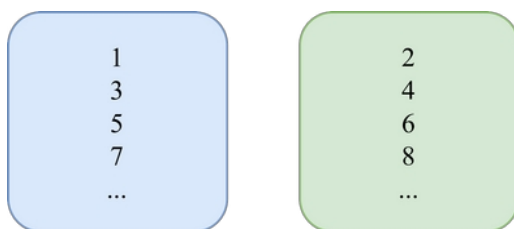
- Линейный RAID (RAID -1)
  - Составной том — простое объединение нескольких дисков в общее пространство



1. LINEAR (RAID -1) — линейный или составной, простое объединение дискового пространства из нескольких устройств. Объединяемые разделы могут быть разного размера.

## RAID массивы

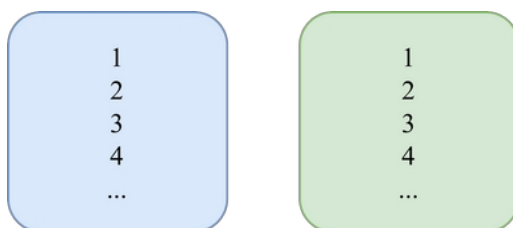
- Чередующийся том (RAID0)
  - Чередование повышает скорость операций чтения-записи
  - Дисковое пространство разбивается на страйпы



2. RAID 0 - для обеспечения высокой производительности дисковой подсистемы, необходимой, например, в системах видеомонтажа;

## RAID массивы

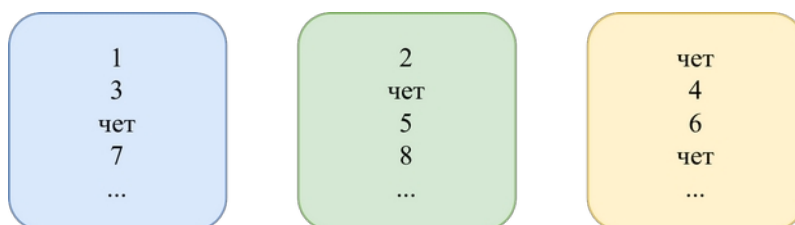
- Зеркальный том (RAID1)
  - Данные дублируются на дисках
  - Можно повысить скорость считывания данных



3. RAID 1 - для обеспечения зеркалирования дисков;
4. RAID 2 - при необходимости использования параллельного доступа с коррекцией ошибок (ECC);
5. RAID 3, обеспечивающий эффективный последовательный доступ к данным и механизм контроля четности с помощью функции XOR;
6. RAID 4, состоящий из нескольких дисков с данными и общего диска для контроля четности с помощью функции XOR;

## RAID массивы

- Чередующийся том с четностью (RAID5)
  - Повышенная отказоустойчивость при меньших затратах чем в зеркале



7. RAID 5, состоящий из нескольких дисков, на каждом из которых хранятся как

## Глава 8. Программные системы хранения

данные, так и контрольные XOR суммы;

8. RAID 10 с чередованием зеркальных наборов дисков;

9. RAID 0/1 - зеркальная схема чередующихся наборов.

Наиболее часто используются RAID массивы 0, 1 и 5 уровней.

### 8.2.2. Программная реализация RAID в Linux.

#### RAID массивы



- Управление программными массивами происходит через драйвер md
- Тип раздела для RAID следует устанавливать в `fd`
- Управление массивами осуществляется командой `mdadm`

Linux поддерживает программную реализацию RAID для следующих уровней (man (8) `mdadm`):

1. линейный режим RAID;
2. RAID 0;
3. RAID 1;
4. RAID 4;
5. RAID 5;
6. RAID 6;
7. RAID 10;
8. MULTIPATH, FAULTY и CONTAINER.

Программная поддержка реализована в ядре с помощью драйвера multiple device driver (`md`).

Для управления RAID массивами применяется программа `mdadm`.

RAID массива может быть собран из разделов и жестких дисков.

При создании RAID из разделов следует устанавливать тип раздела `fd`.

Во время создания RAID массива на устройства записывается специальный блок метаданных, который содержит полное описание массива. Этот блок используется для восстановления конфигурации RAID после перезагрузки ОС или после переноса устройств на другой компьютер.

Метаданные записываются в конец раздела



## Глава 8. Программные системы хранения

mdadm имеет следующие режимы работы:

1. `--create` — создание нового RAID массива;
2. `--assemble` — сборка предварительно созданного массива;
3. `--build` — создание и сборка массива без записи метаданных на диски;
4. `--manage` — управление существующими массивами;
5. `--monitor` — мониторинг работы («здоровья») массива;
6. `--grow` — изменение размеров массива (добавление или удаление членов массива и изменение дискового пространства, которое используется на членах массива);
7. `--incremental` — добавление нового устройства в массив;
8. `--auto-detect` — стартует в ядре активацию любых массивов с автоопределением (этот режим работает не во всех конфигурациях);
9. режим Misc запуск mdadm без выше указанных опций, используется для управления и мониторинга массивами и устройствами.

mdadm поддерживает работу с конфигурационным файлом.

Использование конфигурационного файла не обязательно, но он упрощает администрирование RAID.

По умолчанию предполагается, что конфигурационный называется `/etc/mdadm.conf`.

В файле описываются устройства, под управлением RAID. Ключевое слово `DEVICE`. Массивы, которые используются в системе, описываются ключевым словом `ARRAY`

### **Пример:**

```
DEVICE      /dev/sda1 /dev/sdb1 /dev/sdc1 /dev/sdd1
ARRAY       /dev/md0 devices=/dev/sda1,/dev/sdb1
ARRAY       /dev/md1 devices=/dev/sdc1,/dev/sdd1
```

*В примере показано описание устройств, из которых можно собирать массивы, а также имена имеющихся массивов и устройства, на которых они располагаются. Остальные параметры массивов находятся в метаданных массивов.*

Для создания записи о массиве в конфигурационном файле можно, после создания RAID массива, воспользоваться командой:

```
mdadm --detail --scan >> /etc/mdadm.conf
```

*Не забудьте предварительно создать строку с описанием устройств в файле `/etc/mdadm.conf`.*

Последующая «сборка» массивов описанных в конфигурационном файле осуществляется командой:

```
mdadm -A -s
```

*Эта команда запускается на этапе инициализации системы.*

Опция `--create` предназначена для создания новых массивов.

### **Приер:**

*Создание зеркала:*

## Глава 8. Программные системы хранения

```
mdadm --create /dev/md0 --level=1 -n 2 /dev/hda5 /dev/hdb6
```

*Создание RAID5 с размером полосы чередования 128kB.*

```
mdadm -Cv /dev/md1 -l5 -n3 -c128 /dev/sdb1 /dev/sdc1 /dev/sdd1
```

Зеркальный том можно создать из существующего раздела, без уничтожения данных. Обязательным условием является наличие свободного места на разделе.

Устройство, которое содержит данные для зеркалирования, необходимо поставить первым в команде по созданию зеркала.

Перед созданием зеркала из существующего раздела, рекомендуется предварительно уменьшить размер файловой системы, например утилитой `resize2fs`. Иначе запись метаданных массива в раздел может привести к потере данных.

Создать RAID массивы с чередованием без уничтожения данных на разделе не возможно.

Просмотр состояния массива осуществляется командой:

```
mdadm --detail /dev/md0
/dev/md0:
    Version : 0.90
  Creation Time : Sun Apr 25 17:53:42 2010
    Raid Level : raid1
    Array Size : 248896 (243.10 MiB 254.87 MB)
  Used Dev Size : 248896 (243.10 MiB 254.87 MB)
    Raid Devices : 2
  Total Devices : 2
Preferred Minor : 0
  Persistence : Superblock is persistent

    Update Time : Sun Apr 25 17:53:58 2010
      State : clean, resyncing
  Active Devices : 2
Working Devices : 2
Failed Devices : 0
Spare Devices : 0

Rebuild Status : 72% complete

    UUID : 259fd1fc:40aac815:0ef6a80b:2433f718
  Events : 0.12

    Number Major Minor RaidDevice State
       0       3       5        0    active sync  /dev/hda5
       1       3       6        1    active sync  /dev/hdb6
```

Замена рабочего диска в массиве, который эту операцию поддерживает:

### **Пример:**

```
# mdadm /dev/md0 --fail /dev/hdb6 --remove /dev/hdb6 --add /dev/hdb7
mdadm: set /dev/hdb6 faulty in /dev/md0
mdadm: hot removed /dev/hdb6
mdadm: added /dev/hdb7
```