

## Урок 4. Память

1. Что такое блок в файловых системах?
2. Что такое inode (айнода)?
3. Что такое суперблок?
4. Сколько primary разделов может содержать MBR?

Рекомендуется также посмотреть видео про базовые команды Linux:

<https://youtu.be/Wu6RTLc6Jpc>

*\*Практическое задание (выполнять по возможности и только на "тестовом" диске, так как выполнение может повлечь потерю всех данных)*

- 1) С помощью команд fdisk, mkfs, mount создать логический раздел /dev/sdb1 с файловой системой ext4 и смонтировать ее в каталог /mnt/01
- 2) С помощью команд fdisk, mkfs, mount создать логический раздел /dev/sdb2 с файловой системой ext4 и смонтировать ее в каталог /mnt/02
- 3) Создать любые тестовые файлы в обоих каталогах
- 4) Увеличить размер /dev/sdb1 (использовать команды fdisk, resize2fs). Сохранились ли файлы на /mnt/02? Сохранились ли файлы в /mnt/01?

### 1. Что такое блок в файловых системах?

**Файловые системы (ФС)** определяют способ хранения данных.

Для внутреннего хранения файла ФС разбивает хранилище на блоки:

*В UNIX – блок (мин. 2 сектора), в Windows блок – это кластер.*

Традиционным размером блока ранее были 512 байт, но на сегодняшний день более актуальное значение - 4 килобайта.

**Блоком** называется группа секторов после форматирования диска или раздела сектора на диске, которые разделены на небольшие группы. Размер блока может быть разным и задается как параметр ключа команды форматирования, например:

- **mkfs -t ext4 -b 4096 /dev/sdb**  
*\* где ключ -b задает размер блока в байтах, в данном случае размер блока будет 4096 байт*

Размер блока может быть разным. Это зависит от типа файловой системы:

- Ext2 — 1Кб, 2Кб, 4Кб, 8Кб
- Ext3 — 1Кб, 2Кб, 4Кб, 8Кб
- Ext4 — от 1Кб до 64Кб

При выборе размера блока нужно учесть ряд моментов:

- Максимальный размер файла
- Максимальный размер файловой системы
- Производительность

Размер блока влияет на скорость чтения/записи с диска. Представим себе файл размеров в несколько сот мегабайт, который считывается с диска блоками по 1Кб. Тот же файл будет считываться быстрее, если размер блока файловой системы будет 4Кб или 8Кб. Поэтому при форматировании имеет смысл задать блок большего размера, если планируется использовать файлы большого размера. В случае хранения небольших файлов целесообразно использовать блоки минимального размера

Ядро Linux работает с размером блока файловой системы, а не с размером сектора диска (обычно 512 байт). Важно понимать, что размер блока файловой системы не может быть меньше размера сектора диска и всегда будет кратным ему. Также ядро ожидает, что размер блока файловой системы будет меньше или равно размеру системной страницы

Размер системной страницы можно увидеть выполнив команду:

- `# getconf PAGE_SIZE`
- 4096

Файловые системы бывают **журналируемые** и **нежурналимые**.

Практически все современные ФС относят к общей группе **журналируемых (journal)**: ext3/ext4, NTFS, HFSX, Btrfs и др., так как они ведут учет вносимых изменений в отдельном логе (журнале) и сверяются с ним в случае сбоя при выполнении дисковых операций. Степень подробности ведения журналов и отказоустойчивость у этих файловых систем разные.

К **нежурналируемым (no journal)** – относятся: ext2, FAT16, FAT32

Так например, **Ext3** (Unix/Linux) поддерживает три режима ведения журнала: с обратной связью, упорядоченный и полное журналирование:

- 1-й режим подразумевает запись только общих изменений (метаданных), выполняемую асинхронно по отношению к изменениям самих данных.
- 2-й режим выполняется та же запись метаданных, но строго перед внесением любых изменений.
- 3-й режим эквивалентен полному журналированию (изменений как в метаданных, так и в самих файлах).

Журналирование в **NTFS** (Windows) похоже на второй режим ведения лога в ext3.

В журнал записываются только изменения в метаданных, а сами данные в случае сбоя могут быть утеряны. В NTFS дополнительно периодически создаются контрольные точки, которые гарантируют выполнение всех отложенных ранее дисковых операций. Контрольные точки не имеют ничего общего с точками восстановления, а являются просто служебными записями в логе.

Использование файловых систем лимитировано их поддержкой на уровне ОС. Например, Windows не понимает ext2/3/4 и HFS+, а использовать их порой надо. Сделать это можно, добавив соответствующий драйвер.

Запись на диск производится в следующей последовательности:

- 1) резервирование ФС (Reservation FS)
- 2) Запись (Write)
- 3) Метаданные (meta information)
- 4) Имя файла (Name)

ФС:

Файловая система	Максимальный размер тома	Предельный размер одного файла	Предельное число файлов и/или каталогов	Сжатие данных в фоне	Шифрование данных в фоне	Предельное число файлов и/или каталогов
FAT16	2 Гб секторами по 512 байт или 4 Гб кластерами по 64 Кб	2 Гб	-	-	-	-
FAT32	8 Тб секторами по 2 Кб	4 Гб ( $2^{32} - 1$ байт)	65460	нет	нет	до 32 подкаталогов с CDS
exFAT	$\approx 128$ Пб ( $2^{32}-1$ кластеров по $2^{25}-1$ байт) теоретически / 512 Тб из-за сторонних ограничений	16 Эб ( $2^{64} - 1$ байт)	2796202 в каталоге	нет	нет	32760 символов Unicode, но не более 255 символов в каждом элементе
NTFS (Win)	256 Тб кластерами по 64 Кб или 16 Тб кластерами по 4 Кб	16 Тб (Win 7) / 256 Тб (Win 8)	$2^{32}-1$	да	да	32760 символов Unicode, не более 255 символов в каждом элементе
HFS+	8 Эб ( $2^{63}$ байт)	8 Эб	$2^{32}-1$	да	да	отдельно не ограничивается
APFS	8 Эб ( $2^{63}$ байт)	8 Эб	$2^{63}$	да	да	отдельно не ограничивается
Ext3 (Unix/Linux)	32 Тб (теоретически) / 16 Тб кластерами по 4 Кб (из-за ограничений утилит e2fs programs)	2 Тб (теоретически) / 16 Гб у старых программ	-	нет	нет	отдельно не ограничивается
Ext4 (Unix/Linux)	1 Эб (теоретически) / 16 Тб кластерами по 4 Кб (из-за ограничений утилит e2fs programs)	16 Тб	4 млрд.	нет	да	отдельно не ограничивается
F2FS	16 Тб	3,94 Тб	-	нет	да	отдельно не ограничивается
BTRFS	16 Эб ( $2^{64} - 1$ байт)	16 Эб	-	да	да	$2^{17}$ байт

Разрядность памяти:

Способность процессора работать с тем или иным объемом памяти определяется его разрядностью. Максимальный размер памяти, адресуемый неким регистром  $X$  с битностью  $Y$ , вычисляется как  $2^Y$  байт, объем растет экспоненциально:

- 4-битный регистр адресует  $2^4 = 16$  байт
- 8-битный регистр адресует  $2^8 = 256$  байт, что равно 1 Гигабайт.
- 32-битный регистр адресует  $2^{32} = 4\,294\,967\,296$  байт, что равно порядка 4 Гигабайт.
- 64-битный регистр адресует  $2^{64} = 18\,446\,744\,073\,709\,551\,616 = 16\,777\,216$  Терабайт.

\* 1 байт = 8 бит (1 октет); 1 Кбайт = 1024 байт, 1 Мбайт = 1024 Кбайт, 1 Гбайт = 1024 Мбайт, ...

## 2. Что такое inode (айнода)?

Inode (i-нод) - это структура данных в которой хранится информация о файле или директории в ФС. Например в Ext4 у файла есть не только само его содержимое но и метаданные, такие как:

- имя,
- дата создания,
- доступа,
- модификации и права.

У каждого файла есть своя уникальная i-нод и именно здесь указано в каких блоках находятся данные файла. По умолчанию размер одного блока равен 4092 байта. В начале раздела расположен суперблок, в котором находятся метаданные всей ФС, а к ним идут несколько зарезервированных блоков, а затем размещена таблица i-нод и только после неё блоки с данными. Таким образом, все i-нод размещены в начале раздела диска.

Вывод команды **mke2fs /dev/sdb**

Например:

- Размер блока 4096 байт
- 800 блоковых групп
- 32 768 блоков в группе (8\*4096)

## 3. Что такое суперблок?

Суперблок – это основной элемент (блок) файловой системы Ext(2|3|4), в котором хранятся метаданные ФС.

Аналогично иноду (i-нод), где хранятся метаданные о файлах, суперблок хранит метаданные о ФС. Если вдруг суперблок поврежден, то не возможно будет примонтировать файловую систему. Обычно при загрузке система проверяет суперблок и при необходимости исправляет его, что в результате приводит к корректному монтированию ФС.

Некоторые данные, которые хранятся в суперблоке:

- Количество блоков в файловой системе
- Количество свободных блоков в файловой системе
- Количество i-нод в блоковой группе
- Блоки в блоковой группе
- Количество запусков файловой системы со времени последней проверки fsck
- UUID файловой системы
- Состояние файловой системы (была ли корректно размонтирована, обнаруженные ошибки и т.д.)
- Тип файловой системы
- Операционная система в которой была отформатирована данная файловая система
- Время последнего монтирования
- Время последней записи

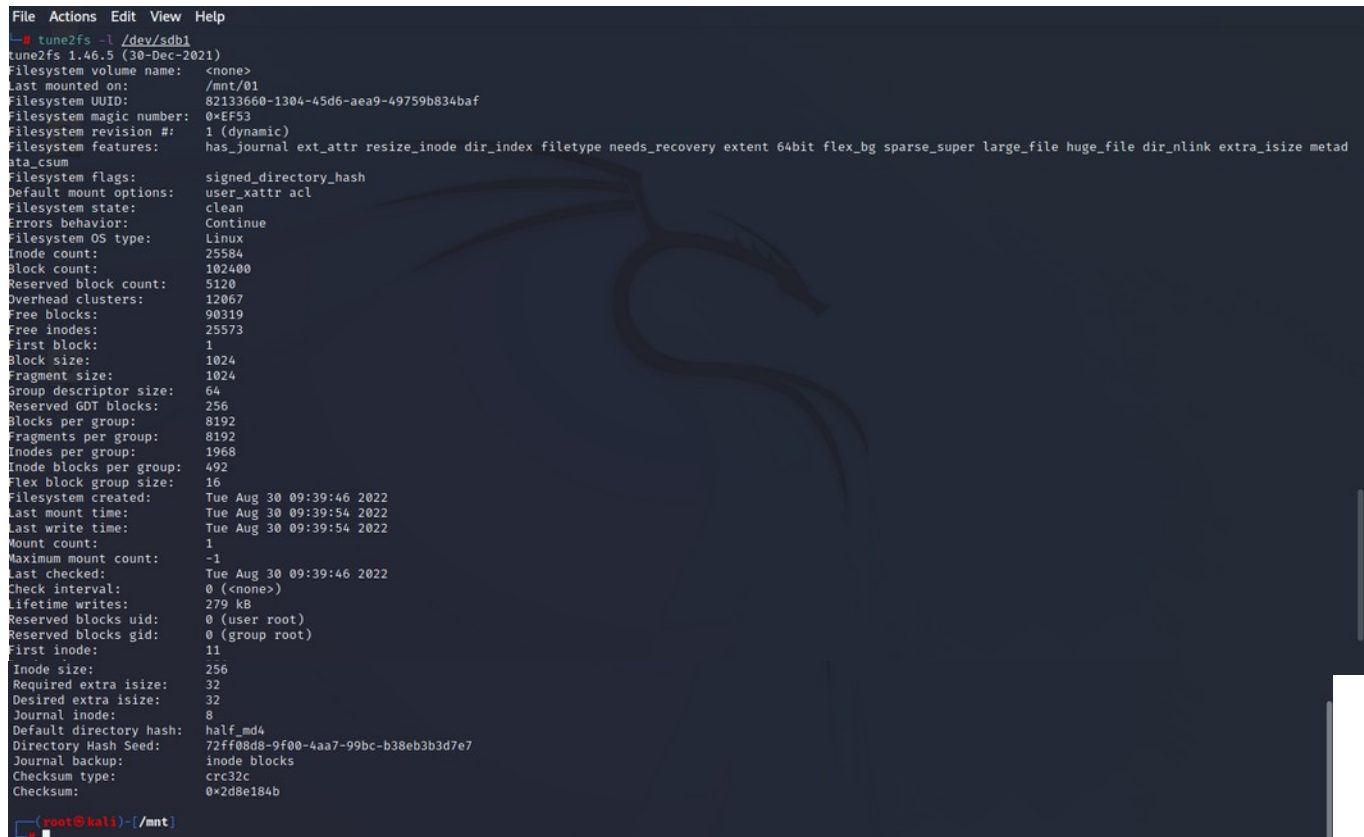
Основная копия суперблока хранится в самой первой группе блоков. Она названа основной, потому что считывается системой в процессе монтирования файловой системы. Так как отсчет блоковых групп начинается с 0, то что суперблок хранится в начале блоковой группы 0. Суперблок весьма критичен для ФС, поэтому в каждой блоковой группе есть копии суперблока и поврежденный суперблок может быть восстановлен, когда это будет необходимо.

Также данной командой (помимо *mke2fs*) можно посмотреть информацию о суперблоке:

- `dumpe2fs /dev/sdb1 | grep -i superblock`
- `dumpe2fs /dev/sdb1 | grep -i superblock`

Проверить и восстановить поврежденный суперблок можно, используя следующие команды:

- `fsck.ext4 -v /dev/sdb`
- **`dumpe2fs /dev/sdb | grep -i superblock` или `mke2fs -n /dev/sdb`**
- `e2fsck -b 819200 /dev/sdb`
- `mount -o -sb=819200 /dev/sdb /mnt`
- `lsblk`
- **`tune2fs -l /dev/sdb1`**



```
File Actions Edit View Help
-# tune2fs -l /dev/sdb1
tune2fs 1.46.5 (30-Dec-2021)
Filesystem volume name: <none>
Last mounted on: /mnt/01
Filesystem UUID: 82133660-1304-45d6-aea9-49759b834baf
Filesystem magic number: 0xEF53
Filesystem revision #: 1 (dynamic)
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 meta
data_csum
Filesystem flags: signed_directory_hash
Default mount options: user_xattr acl
Filesystem state: clean
Errors behavior: Continue
Filesystem OS type: Linux
Inode count: 25584
Block count: 102400
Reserved block count: 5120
Overhead clusters: 12067
Free blocks: 90319
Free inodes: 25573
First block: 1
Block size: 1024
Fragment size: 1024
Group descriptor size: 64
Reserved GDT blocks: 256
Blocks per group: 8192
Fragments per group: 8192
Inodes per group: 1968
Inode blocks per group: 492
Flex block group size: 16
Filesystem created: Tue Aug 30 09:39:46 2022
Last mount time: Tue Aug 30 09:39:54 2022
Last write time: Tue Aug 30 09:39:54 2022
Mount count: 1
Maximum mount count: -1
Last checked: Tue Aug 30 09:39:46 2022
Check interval: 0 (<none>)
Lifetime writes: 279 kB
Reserved blocks uid: 0 (user root)
Reserved blocks gid: 0 (group root)
First inode: 11
Inode size: 256
Required extra isize: 32
Desired extra isize: 32
Journal inode: 8
Default directory hash: half_md4
Directory Hash Seed: 72ff08d8-9f00-4aa7-99bc-b38eb3b3d7e7
Journal backup: inode blocks
Checksum type: crc32c
Checksum: 0x2d8e184b

(root@kali)~[/mnt]
```

```

(root@kali)-[~]
# dumpe2fs /dev/sdb | grep -i superblock
dumpe2fs 1.46.5 (30-Dec-2021)
dumpe2fs: Bad magic number in super-block while trying to open /dev/sdb
Found a dos partition table in /dev/sdb
Couldn't find valid filesystem superblock.

(root@kali)-[~]
# dumpe2fs /dev/sdb1 | grep -i superblock
dumpe2fs 1.46.5 (30-Dec-2021)
Primary superblock at 1, Group descriptors at 2-2
Backup superblock at 8193, Group descriptors at 8194-8194
Backup superblock at 24577, Group descriptors at 24578-24578
Backup superblock at 40961, Group descriptors at 40962-40962
Backup superblock at 57345, Group descriptors at 57346-57346
Backup superblock at 73729, Group descriptors at 73730-73730

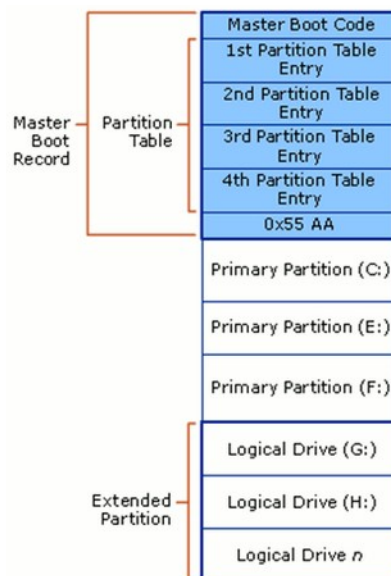
(root@kali)-[~]
# dumpe2fs /dev/sdb2 | grep -i superblock
dumpe2fs 1.46.5 (30-Dec-2021)
Primary superblock at 1, Group descriptors at 2-3
Backup superblock at 8193, Group descriptors at 8194-8195
Backup superblock at 24577, Group descriptors at 24578-24579
Backup superblock at 40961, Group descriptors at 40962-40963
Backup superblock at 57345, Group descriptors at 57346-57347
Backup superblock at 73729, Group descriptors at 73730-73731

```

#### 4. Сколько primary разделов может содержать MBR?

Диск **MBR** может содержать до четырех стандартных (первичных) разделов.

MBR использует 32-битную адресацию пространства, поэтому можно работать только с дисками размером до 2 ТБ, так как на описание количества секторов в разделе отводится 4 байта, следовательно количество секторов ограничено величиной 232, где степень — это количество бит описания (4 байта = 4 x 8 бит = 32 бита). Поскольку размер сектора равен 512 бит, то максимальный размер раздела, который можно описать в таблице разделов MBR, составляет 232 x 512 = 2 ТБ.





**Задания:**

1) С помощью команд **fdisk**, **mkfs**, **mount** создать логический раздел **/dev/sdb1** с файловой системой **ext4** и смонтировать ее в каталог **/mnt/01**

- **fdisk -l /dev/sdb**
- **fdisk /dev/sdb**  
*m (m for help); p (print the partition table); n (add new partions); p (primary); w (write table to disk and exit);*
- **(umount /dev/sdb)**
- **mkfs.ext4 /dev/sdb1**
- **mkdir /mnt/01**
- **mount /dev/sdb1/ /mnt/01**

2) С помощью команд **fdisk**, **mkfs**, **mount** создать логический раздел **/dev/sdb2** с файловой системой **ext4** и смонтировать ее в каталог **/mnt/02**

**По аналогии с пунктом 1.**

3) Создать любые тестовые файлы в обоих каталогах

- **mkdir test**
- **touch file**

```
Command (m for help): p
Disk /dev/sdb: 2 GiB, 2147483648 bytes, 4194304 sectors
Disk model: VBOX HARDDISK
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: 0x97fc9f8b

Device      Boot  Start    End  Sectors  Size Id Type
/dev/sdb1               2048   206847    204800    100M 83 Linux
/dev/sdb2           206848   616447   409600    200M 83 Linux
/dev/sdb3           616448  4194303  3577856    1.7G 83 Linux

Command (m for help): w
The partition table has been altered.
Calling ioctl() to re-read partition table.
Syncing disks.

(root@kali)~#
(root@kali)~# mkfs.ext4 /dev/sdb1
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 102400 1k blocks and 25584 inodes
Filesystem UUID: 82133660-1304-45d6-aea9-49759b834baf
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729

Allocating group tables: done
Writing inode tables: done
Creating journal (4096 blocks): done
Writing superblocks and filesystem accounting information: done

(root@kali)~#
(root@kali)~# mkfs.ext4 /dev/sdb2
mke2fs 1.46.5 (30-Dec-2021)
Creating filesystem with 204800 1k blocks and 51200 inodes
Filesystem UUID: df4c69ee-9604-4bd8-9074-3f55d4c1ca9e
Superblock backups stored on blocks:
    8193, 24577, 40961, 57345, 73729
```

```
(root@kali)~# mkdir /mnt/01/test_01

(root@kali)~# mkdir /mnt/02/test_02

(root@kali)~# touch /mnt/01/file1

(root@kali)~# touch /mnt/02/file2

(root@kali)~# df -h
Filesystem      Size  Used Avail Use% Mounted on
udev            956M   0    956M   0% /dev
tmpfs           199M  964K  198M   1% /run
/dev/sda1       78G   14G   61G  18% /
tmpfs           991M   0    991M   0% /dev/shm
tmpfs           5.0M   0    5.0M   0% /run/lock
tmpfs           199M   84K  199M   1% /run/user/1000
/dev/sdb1        89M   15K   82M   1% /mnt/01
/dev/sdb2       182M   15K  168M   1% /mnt/02

(root@kali)~# ls /mnt/01
file1  lost+found  test_01

(root@kali)~# ls /mnt/02
file2  lost+found  test_02
```

4) Увеличить размер /dev/sdb1 (использовать команды **fdisk**, **resize2fs**).  
Сохранились ли файлы на /mnt/02? Сохранились ли файлы в /mnt/01?

- **fdisk /dev/sdb1**

Чтобы расширить раздел требуется предварительно удалить информацию о нём.  
Для этого вводим **d** (*delete a partition*) и указываем раздел.

- Удаляем все разделы, так как *sdb1* тожно будет создать только такого же или меньшего размера **d** (*delete partition*) **n** (*new partition*) **w** (*write*)

*При этом удаляется только запись о разделе, сами данные остаются на диске!*

- **resize2fs /dev/sdb1**

```

Disk /dev/sdb: 2 GiB, 2147483648 bytes, 4194304 sectors
Disk model: VBOX HARDDISK
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: 0x97fc9f8b

Device      Boot  Start      End  Sectors  Size Id Type
/dev/sdb1   2048    206847    204800    100M 83 Linux
/dev/sdb2   206848    616447    409600    200M 83 Linux
/dev/sdb3   616448   1230847    614400    300M 83 Linux

Command (m for help): d
Partition number (1-3, default 3): 1
Partition 1 has been deleted.

Command (m for help): d
Partition number (2,3, default 3): 2
Partition 2 has been deleted.

Command (m for help): d
Selected partition 3
Partition 3 has been deleted.

Command (m for help): w
The partition table has been altered.
Failed to remove partition 1 from system: Device or resource busy
Failed to remove partition 2 from system: Device or resource busy
Failed to remove partition 3 from system: Device or resource busy

The kernel still uses the old partitions. The new table will be used at the next reboot.
Syncing disks.

```

```

Disk /dev/sdb: 2 GiB, 2147483648 bytes, 4194304 sectors
Disk model: VBOX HARDDISK
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: 0x97fc9f8b

Device      Boot  Start      End  Sectors  Size Id Type
/dev/sdb1   2048   309247    307200    150M 83 Linux
/dev/sdb2   309248   821247    512000    250M 83 Linux

Command (m for help): w
The partition table has been altered.
Syncing disks.

```

```

(root@kali)~# ls /mnt/02
file2  lost+found  test_02

(root@kali)~# ls /mnt/01
file1  lost+found  test_01

(root@kali)~# resize2fs /dev/sdb1
resize2fs 1.46.5 (30-Dec-2021)
The filesystem is already 102400 (1k) blocks long.  Nothing to do!

(root@kali)~# resize2fs /dev/sdb2
resize2fs 1.46.5 (30-Dec-2021)
The filesystem is already 204800 (1k) blocks long.  Nothing to do!

```

```

(root@kali)~# fdisk /dev/sdb

Welcome to fdisk (util-linux 2.38.1).
Changes will remain in memory only, until you decide to write them.
Be careful before using the write command.

This disk is currently in use - repartitioning is probably a bad idea.
It's recommended to umount all file systems, and swapoff all swap
partitions on this disk.

Command (m for help): p

Disk /dev/sdb: 2 GiB, 2147483648 bytes, 4194304 sectors
Disk model: VBOX HARDDISK
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: 0x97fc9f8b

Device      Boot  Start      End  Sectors  Size Id Type
/dev/sdb1   2048   309247    307200    150M 83 Linux
/dev/sdb2   309248   821247    512000    250M 83 Linux

```