


LPIC-1 101-400 – Lesson 17


104.1 Create partitions and filesystems



Disk Devices

- IDE disks (ATA or PATA) were traditionally designated as: **/dev/hda** (primary master), **/dev/hdb** (primary slave), **/dev/hdc** (secondary master), **/dev/hdd** (secondary slave) and so on.
 - SCSI disks are designated as **/dev/sda**, **/dev/sdb**, **/dev/sdc** and so on. This also includes devices that are not in fact SCSI but they use the SCSI subsystem of the Linux kernel like **SATA**, **SAS**, **USB Storage** and in recent kernels even legacy **IDE** devices!
 - Virtual disks are typically designated as **/dev/vda**, **/dev/vdb** and so on.
- 

Partitions and File Systems

- **Partitions** are logical divisions of a disk to smaller regions
 - **Filesystems** are logical structures on a partition which facilitate in the saving and retrieving data in the form of files
 - Filesystems are attached to mount points which are must be directories
 - There are virtual file systems, like **proc** and **sysfs**, which are not related to a disk or partition, but do in fact exists in memory
- 

Primary, Extended and Logical Partitions

- These concepts apply only for **MBR** (aka MSDOS) partition layouts, not for **GPT**
- **Primary partitions:** These partitions are defined in the MBR sector. It is mandatory to have at least 1 and can be up to 4. Their IDs start from 1 to 4 e.g. **/dev/sda1 – /dev/sda**
- **Extended partition:** it is a type of primary partition (usually the last of the primary partitions) and it serves as a container for logical partitions. It can not host a filesystem! There can be only one in each disk
- **Logical partitions:** they exist inside the extended partition and can be used in case we need more than 4 partitions per disk. There can be up to 12 logical partitions and their IDs start from 5 until 16 e.g. **/dev/sda5 /dev/sda6**

The *GPT* partition layout

- Modern systems support the new **GPT** partition layout which has several advantages over the legacy **MBR**:
 - Up to 128 primary partitions vs 15 in MBR
 - Volume sizes up to 18 PB vs 2TB in MBR
 - Has backup partition table for redundancy
 - Each partition has a unique ID
 - Has some limited backward compatibility with MBR



Filesystem types

- **ext2** second extended filesystem: . This had been the main filesystem used in the early days of Linux. It does not support journaling. The lack of journaling makes it favorable for flash drives (miniSD, USB, SSD) because fewer writes help extend the flash disks lifetime
- **ext3** third extended filesystem: basically it has the same features as ext2 and it is backward compatible with it. The main advantage is the support for **journaling** which allows better reliability and less checks during abrupt system interruptions
- **ext4** fourth extended filesystem: it is backward compatible with the previous two but it supports file sizes up to 16TiB in comparison with 2TiB of **ext2** and **ext3**. It also provides speed improvements and more reliability

Filesystem types

- **xfs**: developed by Silicon Graphics with the goal of large files support. It supports journaling and smooth data transfer. Default on recent CentOS versions
- **vfat (FAT32)**: used on obsolete Windows versions like 95, 98 and ME. Still popular on removable disk devices. The maximum file size is 4GB
- **exFAT**: developed by Microsoft as an improvement over FAT32 with better features like support for files bigger than 4GB
- **Btrfs**: a new generation filesystem with impressive features like copy-on-write, transparent compression, deduplication, raid and lots more. Still in development but some features are production ready

Show partitions and filesystems

- `$ cat /proc/partitions` # show kernel detected partitions
- `$ mount` # show mount points, filesystem types and mount options.
- `$ df -hT` # show mounted filesystem in human-readable form, with percent usage and type (-T)
- `# fdisk -l /dev/sda` # list partitions of `sda` with size and filesystems. Recent versions support GPT



Create partitions with `fdisk`

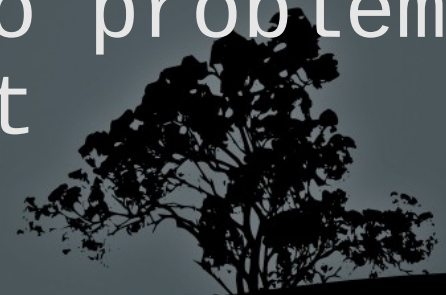
- **fdisk** is interactive and has its own shell. It is used for partition creation
- `# fdisk /dev/sda` # choose disk
/dev/sda
Command (m for help):
- `m` # show list of commands
- `l` # show partition types
- `p` # list partitions of `sda`
- `q` # exit **fdisk** without saving
- `w` # write changes and exit **fdisk**

The `parted` and `gdisk` commands

- **parted** is an advanced partition manager and the first one to support the new GPT partitioning scheme. It also support MBR
- `# parted -l /dev/sda # list partitions of sda with size and filesystems. Supports GPT`
- `# parted /dev/sda # enter in parted's interactive mode for partitioning sda`
- **gdisk** was designed as an alternative to **fdisk** at a time when **fdisk** did not support GPT. **gdisk** supports only **GPT** and it will automatically convert an MBR disk to GPT
- `# gdisk -l /dev/sda # list partition table of sda`

Create partitions with `fdisk`

- **a** # set bootable flag (legacy systems)
- **n** # create new partition
- **d** # delete partitions
- **t** # change partition type
- **v** # verify partition table. You should use it before running **w** to verify that there is no problem in the partition layout



Create partitions with `fdisk`

```
# fdisk /dev/sda # process sda partition table
```

The number of cylinders for this disk is set to 9729.

There is nothing wrong with that, but this is larger than 1024, and could in certain setups cause problems with:

- 1) software that runs at boot time (e.g., old versions of LILO)
- 2) booting and partitioning software from other OSs

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

Command (m for help):



Create filesystems with `mkfs`

- In the system there is a family of command in the form **mkfs.<fstype>** where **fstype** can be one of these: **ext2**, **ext3**, **ext4**, **xfs**, **vfat**, **exfat**, **ntfs** etc
- There is also a generic **mkfs** command where the type of the filesystem has to be defined with the option **-t** e.g.: **mkfs -t ext3**
- For **ext2**, **ext3** and **ext4** filesystems there is also **mke2fs**



Create filesystems with `mkfs`

- `# mkfs -t ext2 /dev/sda3 # = mkfs.ext2, mke2fs`. Create an `ext2` filesystem on the `sda3` partition
- `# mkfs -t ext3 /dev/sda3 # = mkfs.ext3, mke2fs -j`. Create an `ext3` filesystem on the `sda3` partition
- `# mkfs -t ext4 /dev/sda3 # = mkfs.ext4`. Create an `ext4` filesystem on the `sda3` partition

Create filesystems with `mkfs`

- `# mkfs -t vfat /dev/sda3 # = mkfs.vfat`. Create a FAT32 filesystem on the `sda3` partition
- `# mkfs -t xfs /dev/sda3 # = mkfs.xfs`. Create an XFS filesystem on the `sda3` partition



Create filesystems with `mkfs`

Options:

- `-L` # create a label) for `ext2`, `ext3` and `ext4` filesystems
- `-c` # check for corrupted blocks before the creation of the filesystem
- `-j` # create `ext3` journal. Only for `mkfs.ext3` and `mke2fs`
- `-q` # quit mode with minimal output
- `-v` # verbose output



Create swap partitions with `mkswap`

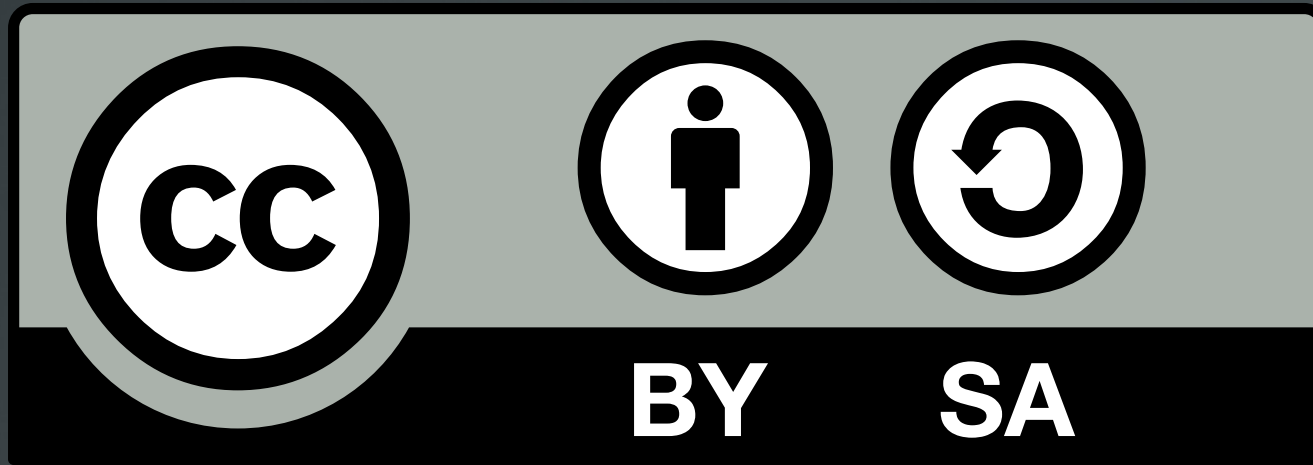
- `# mkswap /dev/sda6 # format a swap partition sda6`

Options:

- `-c # check for corrupted blocks before creation`
- `-L # set label for partition`
- `-p <SIZE> # set page size`



License



The work titled "LPIC-1 101-500 – Lesson 17" by Theodotos Andreou is distributed with the Creative Commons Attribution ShareAlike 4.0 International License.

