Linux File System

Dr. Vimal Kr Baghel (Course Instructor), Assistant Professor
School of Computer Science Engineering & Technology (SCSET)
Bennett University Greater Noida

BENNETT UNIVERSITY
THE TIMES GROUP

Outline

- Introduction to filesystem
- Inode (index node)
- Filesystem utilities
- Q & A

Understanding the basic Linux filesystems

- Filesystem is used to store files and folders on a storage device
- Most Linux distros provide a default filesystem for us at installation time
- Each filesystem implements the **virtual directory structure** on storage devices

BENNETT UNIVERSITY
THE TIMES GROUP

Understanding the basic Linux filesystems

- The original filesystem introduced with the Linux is called the extended filesystem (ext).

- Ext uses virtual directories to handle physical devices and storing data in fixed-length blocks on the physical devices.

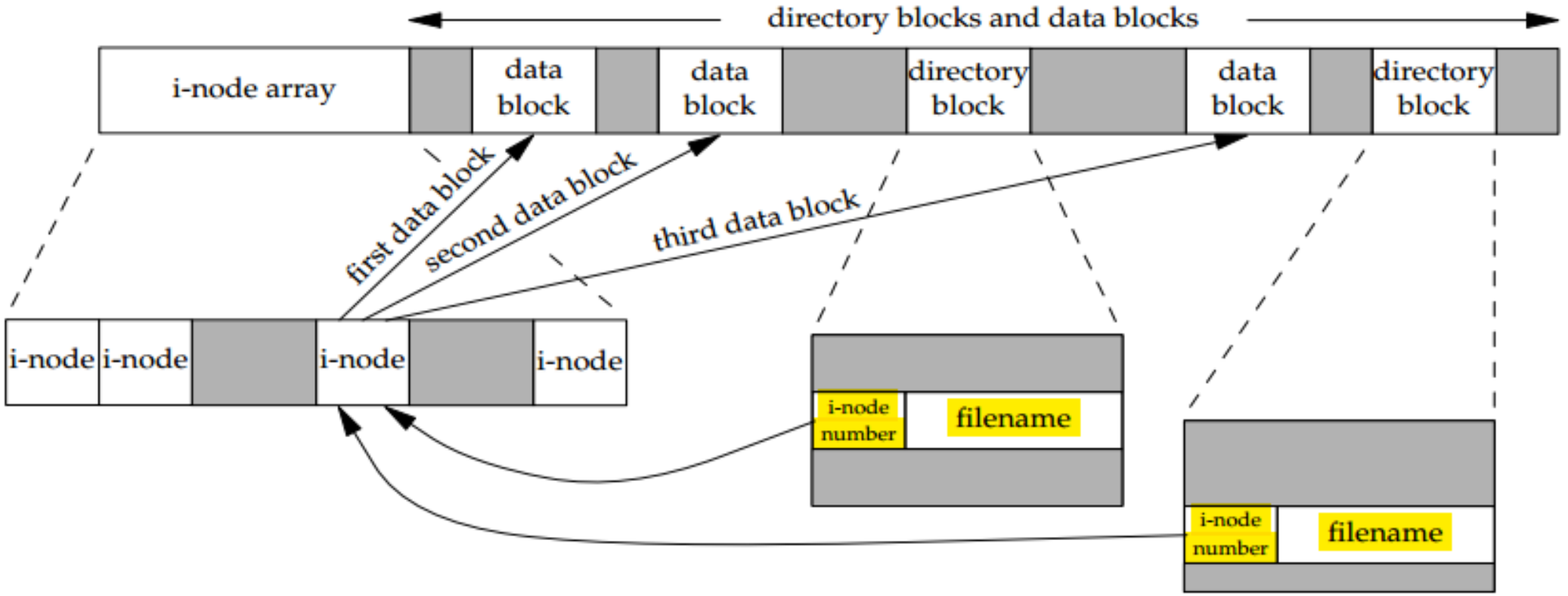
- The ext filesystem uses *inodes* to track information about the files stored in the virtual directory.

inodes

- The inode system creates a separate table on each physical device, called the **inode table, to store file information**.
- Each stored file in the virtual directory has an entry in the inode table.
- The extended part of the name comes from the additional data that it tracks on each file, which consists of:
  - The filename
  - File Size (upto 2GB in ext & upto 3TB to 32 TB in ext2)
  - The owner of the file & the group the file belongs to
  - Access permissions for the file
  - Pointers to each disk block that contains data from the file

Inode

- The internal file representation is done through *inode. Inode* contains the description such as:
  - disk layout of file data,
  - *Owner* of the file,
  - *File access* permissions, and
  - Access times

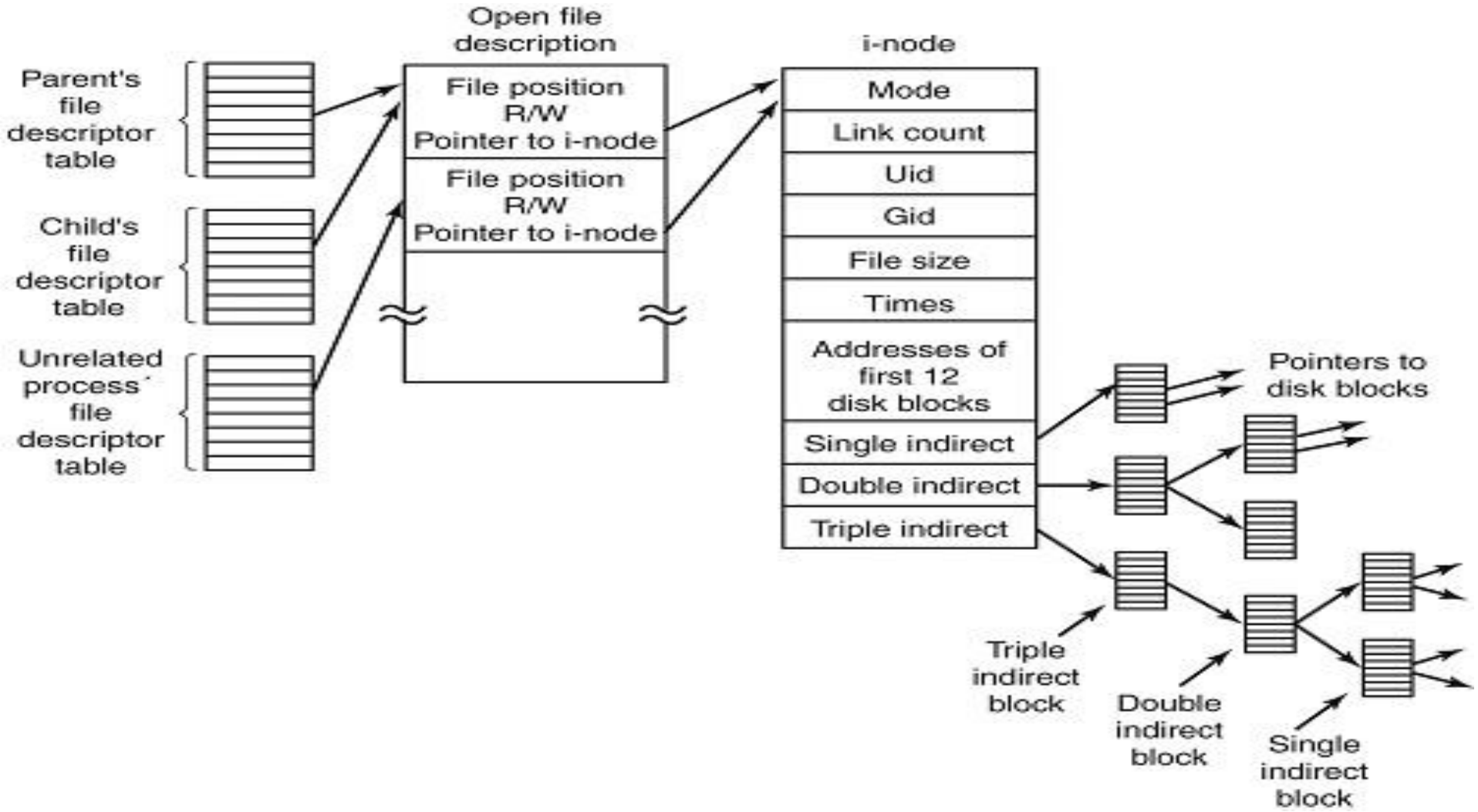


inode

- Each file has one inode, and can have multiple names (link)
- All names map onto its single inode
- Example
  - Open("/home/vimal/e1.sh" , 1)
  - Kernel assigns an unused inode

```
$ ls -i *data_file
296890 data_file 296891
sl_data_file
$
```

Inode structure



A video on Inode Structure

BENNETT UNIVERSITY
THE TIMES GROUP

Inode Structure

Understanding journaling filesystems

BENNETT UNIVERSITY
THE TIMES GROUP

- *Journaling filesystems* provide a new level of safety to the Linux filesystem.
- Instead of writing data directly to the storage device and then updating the inode table, journaling filesystems write file changes into a temporary file (*journal*) first.
- After data is successfully written to the storage device and the inode table, the journal entry is deleted.
- In case of crash/power outage before the data can be written to the storage device, the journaling filesystem just reads through the journal file and processes any uncommitted data left over.

Journaling Methods

Method	Description
Data mode	Both inode and file data are journaled. Low risk of losing data, but poor performance.
Ordered mode	Only inode data is written to the journal, but not removed until file data is successfully written. Good compromise between performance and safety.
Writeback mode	Only inode data is written to the journal, no control over when the file data is written. Higher risk of losing data, but still better than not using journaling.

Working with Linux filesystem

- Creating partitions
  - $ **fdisk /dev/sdb**
  - Unable to open /dev/sdb
  - $

  - $ **sudo fdisk /dev/sdb**
  - [sudo] password for vimal:
  - Device contains neither a valid DOS partition table,
  - nor Sun, SGI or OSF disklabel
  - Building a new DOS disklabel with disk identifier 0xd3f759b5.
  - Changes will remain in memory only
  - until you decide to write them.
  - After that, of course, the previous content won't be recoverable.
  - Warning: invalid flag 0x0000 of partition table 4 will
  - be corrected by w(rite)
  - [...]
  - Command (m for help):

Thanks

Q & A