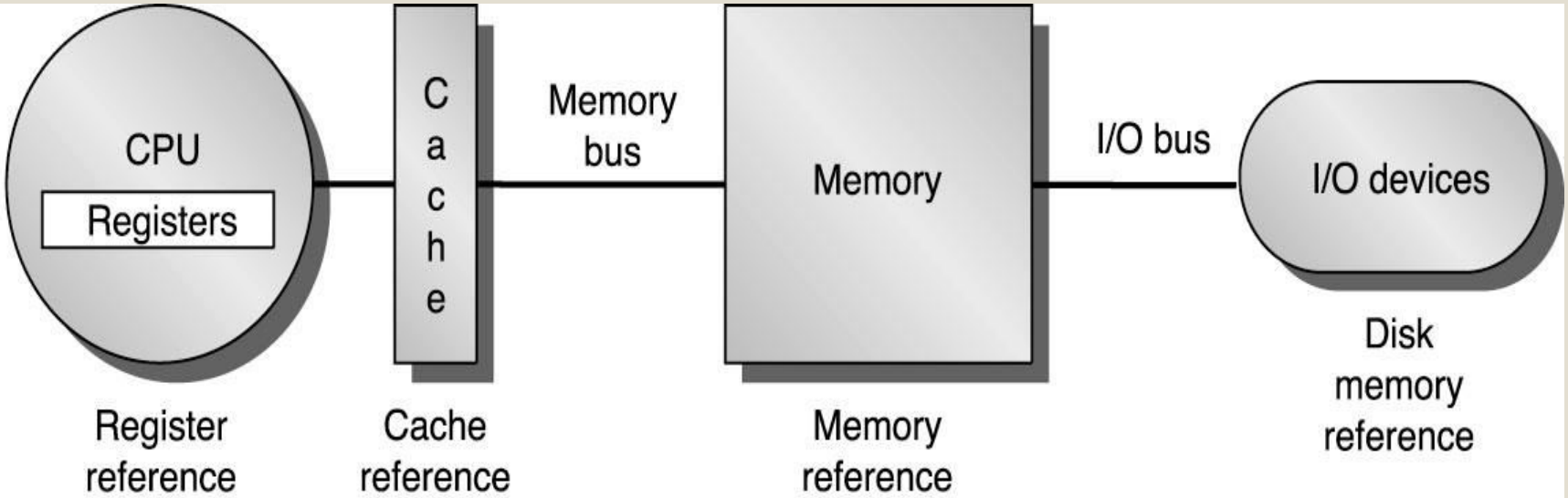
File Systems

OPERATING SYSTEMS COURSE

THE HEBREW UNIVERSITY

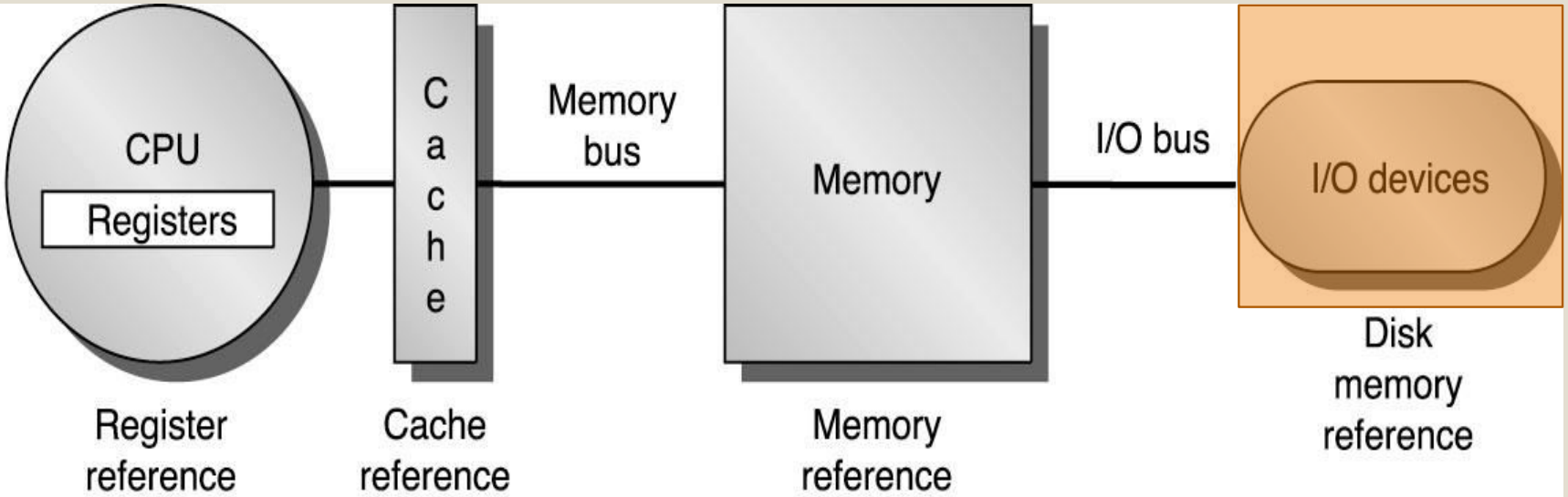
SPRING 2023

Typical Memory Hierarchy



	Register reference	Cache reference	Memory reference	Disk memory reference
Size:	500 bytes	64 KB	512 MB	100 GB
Speed:	0.25 ns	1 ns	100 ns	5 ms
Current Size	2MB	20MB	64GB	12TB

Typical Memory Hierarchy



	Register reference	Cache reference	Memory reference	Disk memory reference
Size:	500 bytes	64 KB	512 MB	100 GB
Speed:	0.25 ns	1 ns	100 ns	5 ms
Current Size	2MB	20MB	64GB	12TB

File

- **A logical unit of information**
  - ADT (abstract data type)
  - Has a name
  - Has content (typically a sequence of bytes).
  - Has metadata/attributes (creation date, size, …)
  - Can apply operations to it (read, rename, …)

- **Which is persistent (non-volatile)**
  - Survives power outage, outlives processes
  - Process can use file, die, then another process can use file

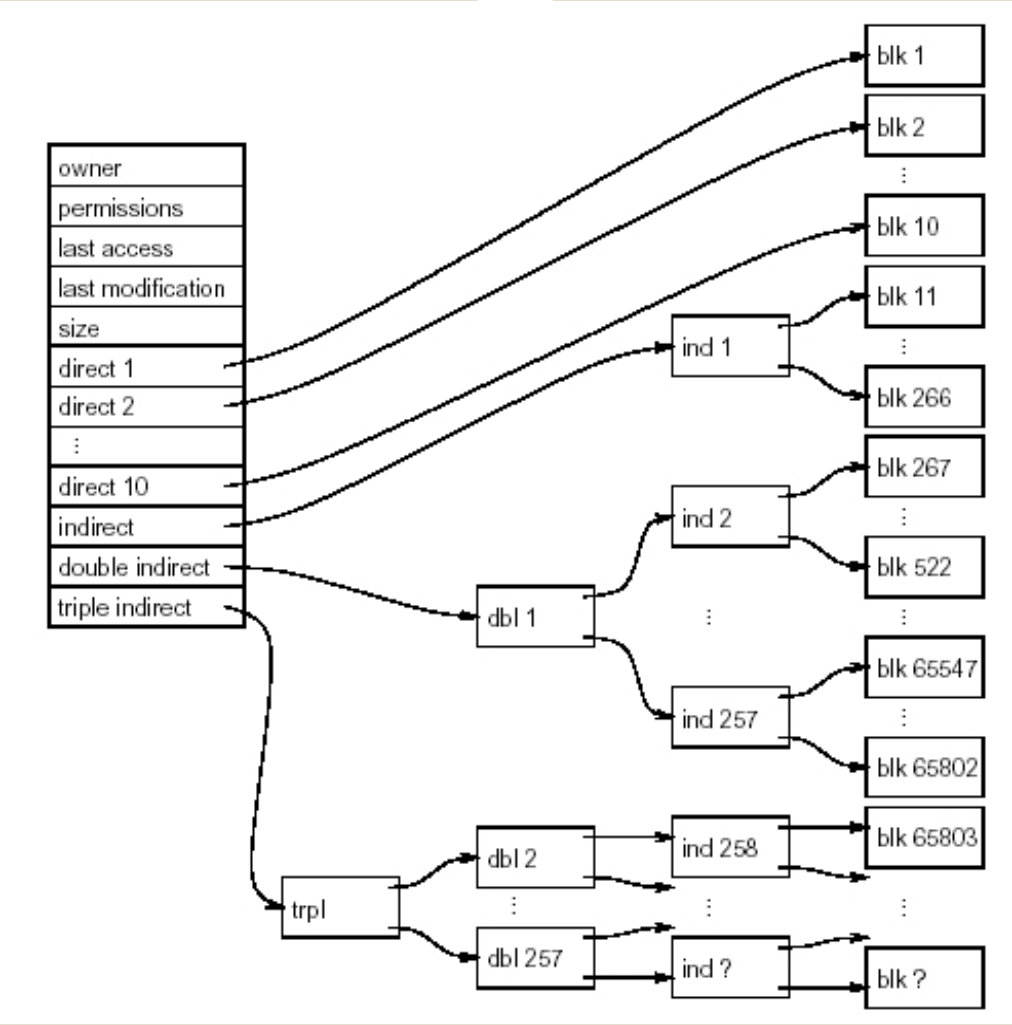
File Metadata

- Size
- Owner
- Permissions
  - Readable? Writable? Executable?
- Timestamps
  - Creation time
  - Last time content\metadata where modified\accessed
- Location
  - On disk (recall that a disk is a “block device”)
  - Where do the file’s blocks reside on disk?
- Type
  - E.g., binary vs. text, or regular file vs. directory

The Unix inode

- inode is a data structure in Unix File System that stores a file (or directory) metadata.
- Includes an hierarchical index of disk blocks where the file is located:
  - Few (~12) *direct* pointers, which list the first blocks of the file (Allows small files)
  - One single *indirect* pointer - points to a whole block of additional direct pointers (Allows medium files)
  - One *double* indirect pointer - points to a block of indirect pointers. (Allows large files)
  - One *triple* indirect pointer - points to a block of double indirect pointers. (Allows huge files)

inode Structure



inode – a more up-to-date example

- Assume 32 bit inodes (each pointer is 4 bytes), blocks are 4096 bytes.
  - The 12 direct pointers then provide access to a maximum of 48 KB.
  - The indirect block contains 1024 additional pointers, for a data of size (4 MB).
  - The double indirect block has 1024 pointers to indirect blocks, so it points to 4GB of data
  - The triple indirect block allow files of 4TB.

Directories

- Directories are stored like files.
  - In metadata (inode): type = directory.
- Their content (data blocks) encodes a mapping of contained file/directory to inode.
  - can be shown using the -i option to the ls command

```
~ $ ls -i /
     4 bin        7063 lib          34 sbin        39 var
     3 dev        7238 linuxrc       1 sys
     9 etc           1 proc         38 tmp
    22 home       7096 root         25 usr
~ $
```

Directories – cont.

- Files doesn't store their names. Only the directories store the names of the files within them.

```
~ $ ls -ila /etc/network
total 3
   17 drwxrwxr-x    1 root     root          151 May 18 06:28 .
    9 drwxrwxr-x    1 root     root          754 May 18 06:28 ..
 7447 drwxr-xr-x    1 root     root           51 May 18 06:28 if-pre-up.d
 7446 drwxr-xr-x    1 root     root           51 May 18 06:28 if-up.d
   18 -rw-r--r--    1 root     root           64 May 18 06:28 interfaces
~ $
```

File inode

Directory content

How do we allocate blocks for files (and directories)?

Superblock

- Manages the allocation of blocks on the file system area.
- This block contains:
  - The size of the file system (FS).
  - A list of free blocks available on the FS.
  - A list of unused inodes
  - And more…
- Using this information it is possible to allocate disk block for saving file data or file metadata.

inode allocations

- inodes' space is allocated when creating a new FS.
- Hence the number of inodes is limited.
- The superblock caches a short list of free inodes. When a process needs a new inode, the kernel can use this list to allocate one.
- When an inode is freed, its location is written in the superblock, but only if there is room in the list.
- If the superblock list of free inodes is empty, the kernel searches the disk and adds other free inodes to its list.

Data Blocks Allocations

- When a process writes data to a file, the kernel must allocate disk blocks from the file system for a direct or indirect block.

- When the kernel wants to allocate a block from the file system, it allocates the next available block in the superblock list.

Storing and Accessing File Data

- Storing data in a file involves:
  - Allocation of disk blocks to the file.
  - Read (not always) and write operations.
  - Optimization - avoiding disk access by caching data in memory.

Opening a File

fd=open(“myfile",R)

- A **file descriptor** (**FD**) is an abstract handle used to access a file or other input/output resources, such as a network socket (next lesson).
- Threads share the file descriptor table (and the file offset).
- Once a FD exists, it will always point to the same file.
- “Everything” is a file in Unix.
- A successful open(“file name”) returns a FD:
  - A nonnegative integer.
  - An index to a per-process array called the “file descriptor table”.

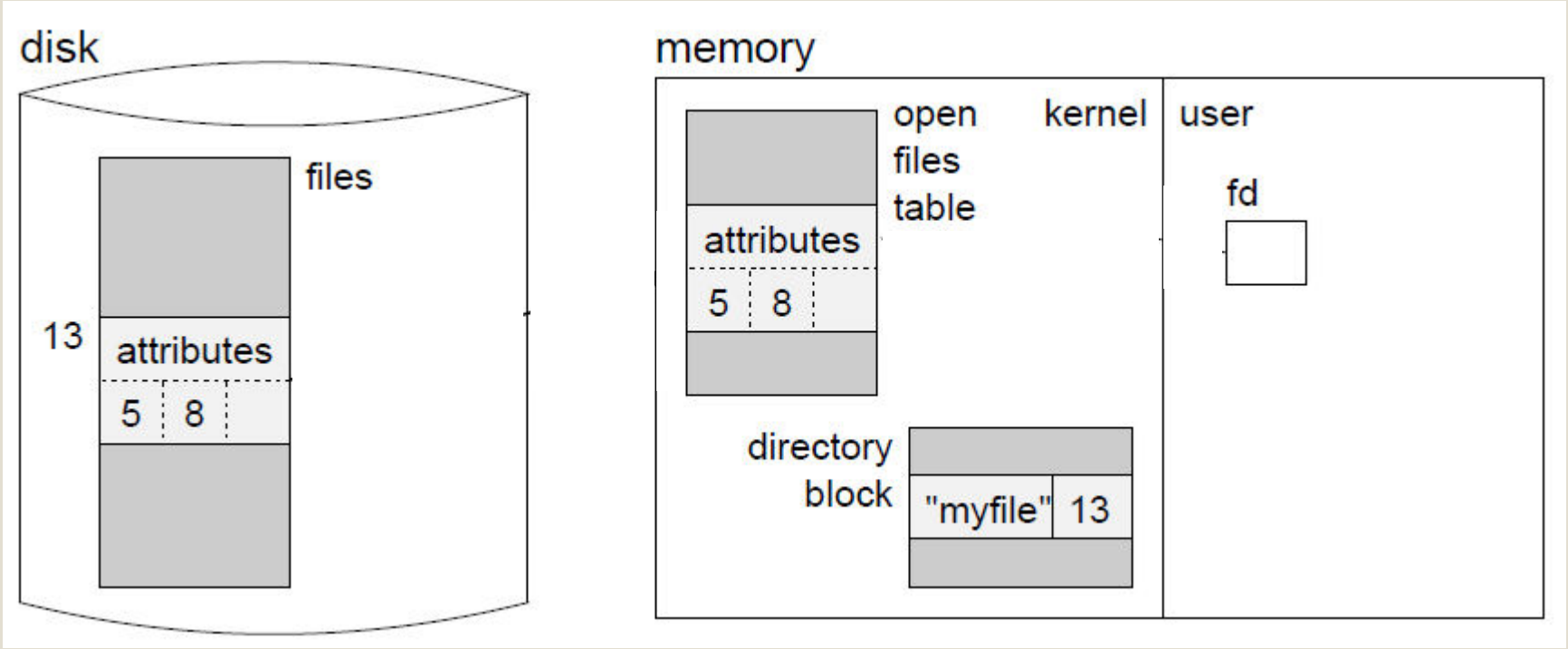
Opening a File – cont.

- Threre are several predefined FDs:

File	File Descriptor	POSIX Symbolic Constant
Standard Input	0	STDIN_FILENO
Standard Output	1	STDOUT_FILENO
Standard Error	2	STDERR

Opening a File – cont.

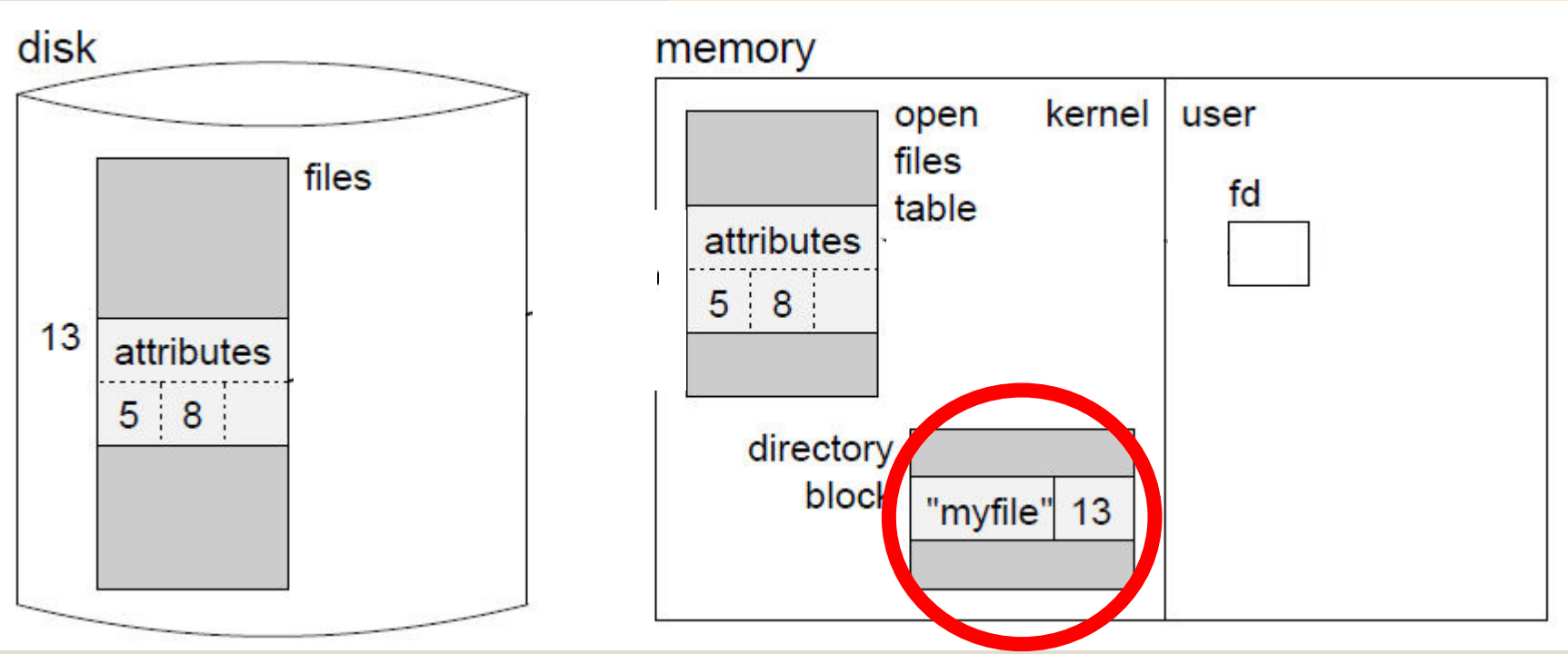
fd=open(“myfile",R)



Opening a File

fd=open(“myfile",R)

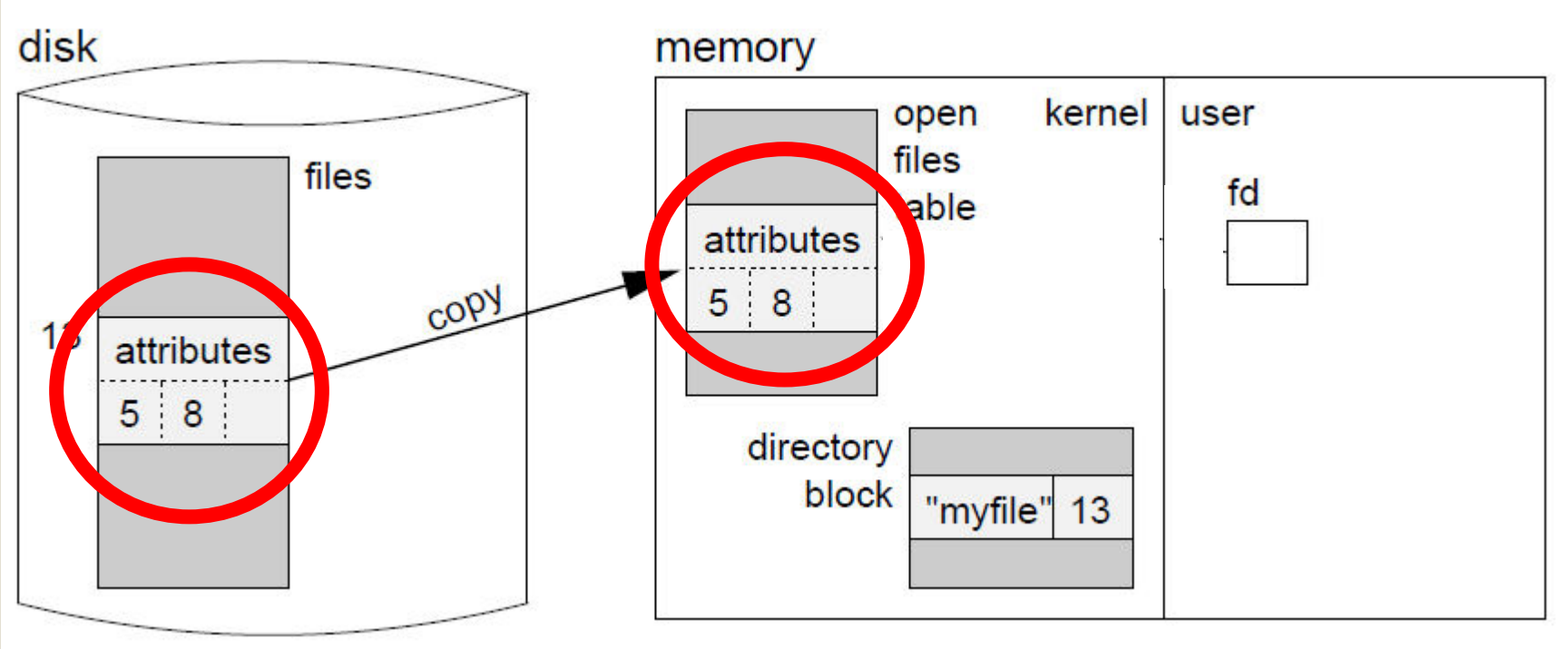
1. FS reads the current directory, and finds that “myfile” is represented internally by entry 13 (inode number) in the list of files maintained on the disk.



Opening a File

fd=open(“myfile",R)

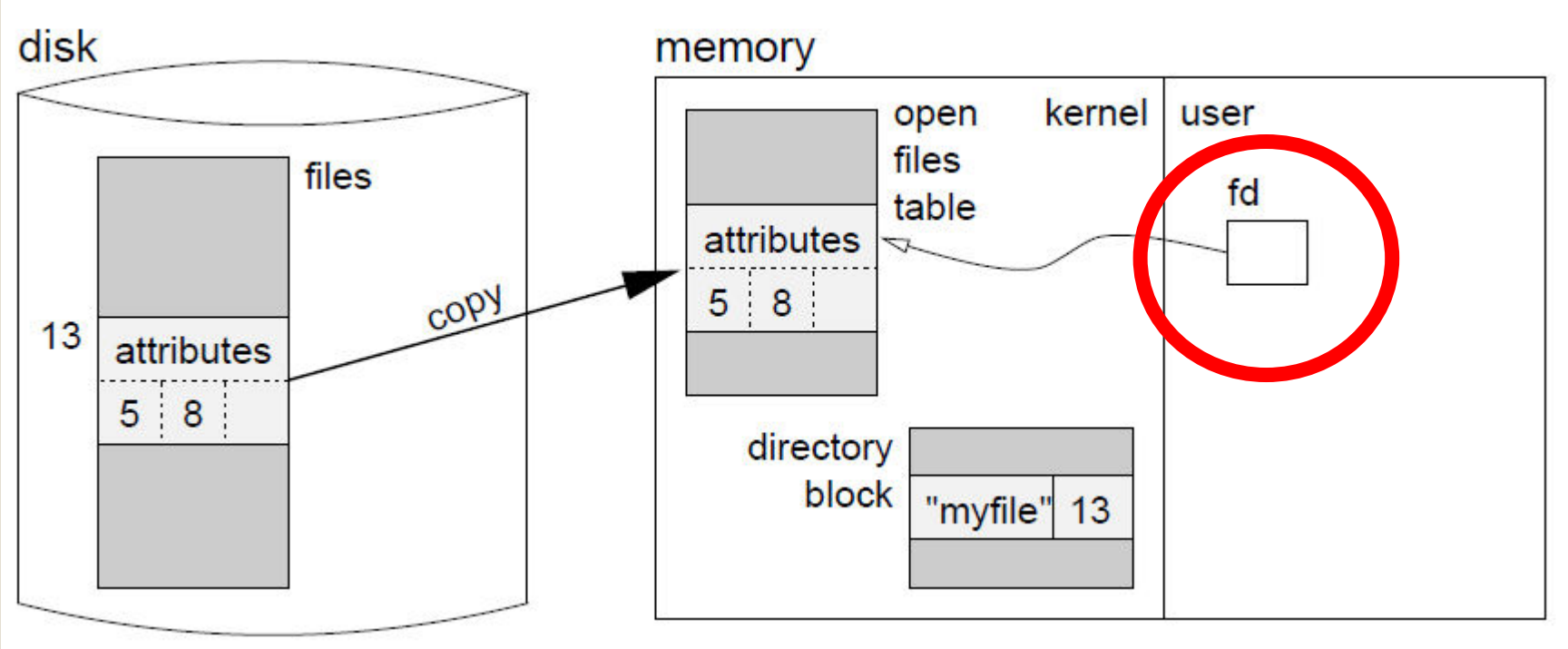
2. Entry 13 from that data structure is read from the disk and copied into the kernel’s open files table.



Opening a File

fd=open(“myfile",R)

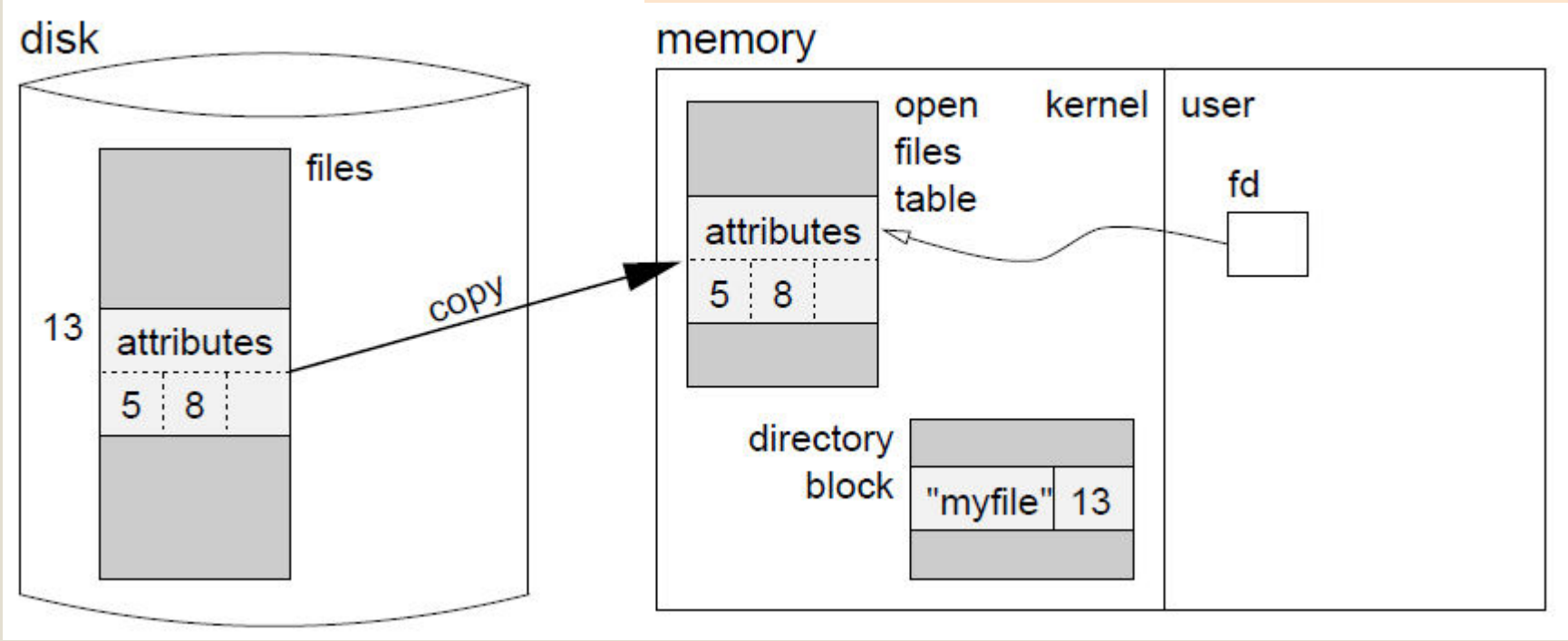
3. User’s access rights are checked and the user’s variable fd is made to point to the allocated entry in the open files table.



Opening a File

fd=open(“myfile",R)

4. FD serves as a handle, telling to which file the user tries to access subsequently by read\write system calls,without allowing the user code actual access to kernel data.

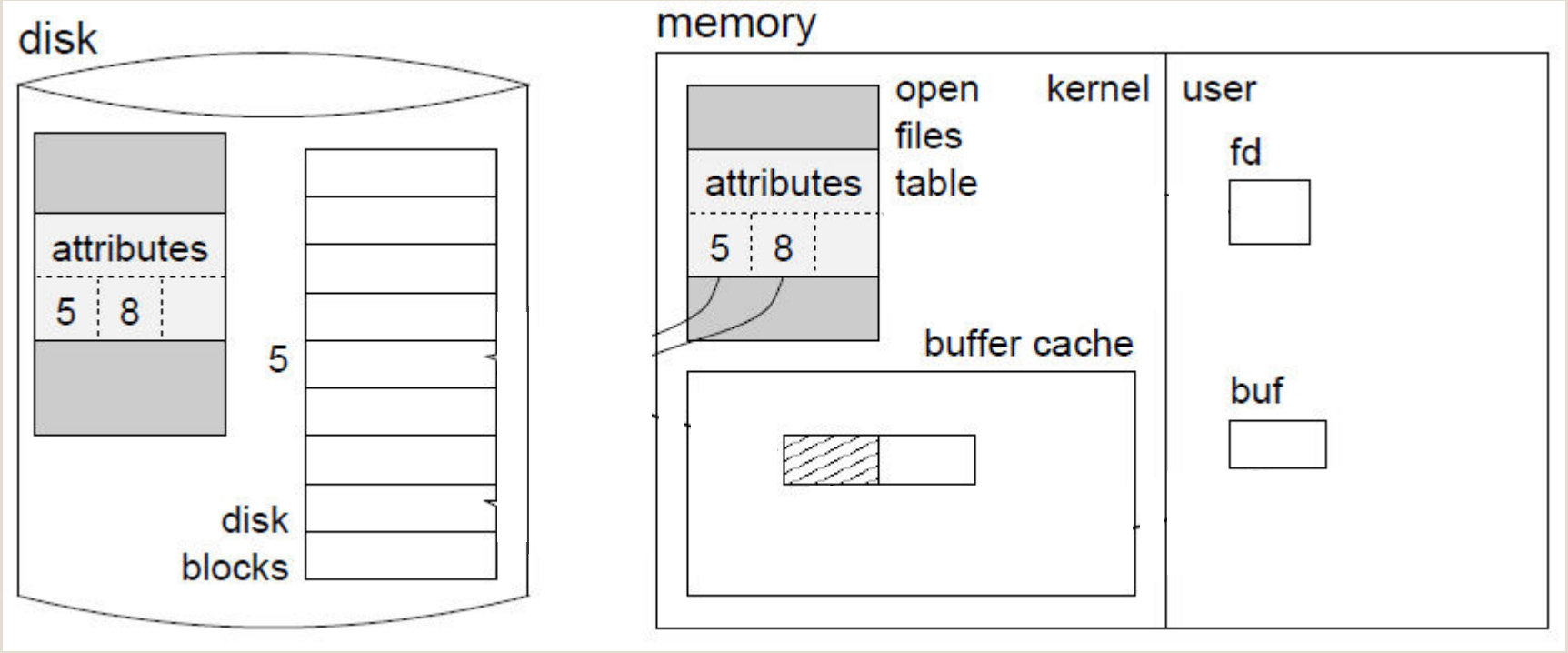


Opening a File

- Why do we need the open file table?
  - check permissions (once)
  - store the offset
- All the operations on the file are performed through the file descriptor.

Reading from a File

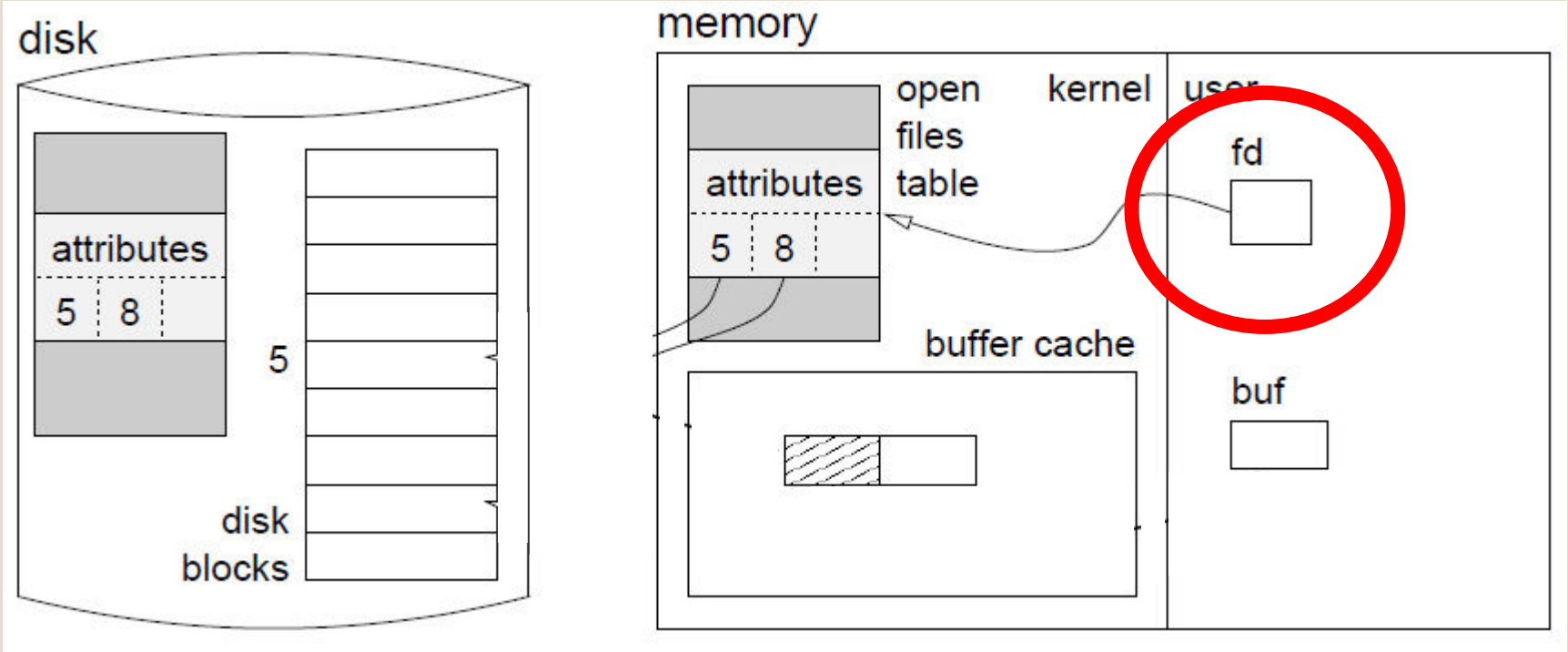
read(fd,buf,100)



Reading from a File

read(fd,buf,100)

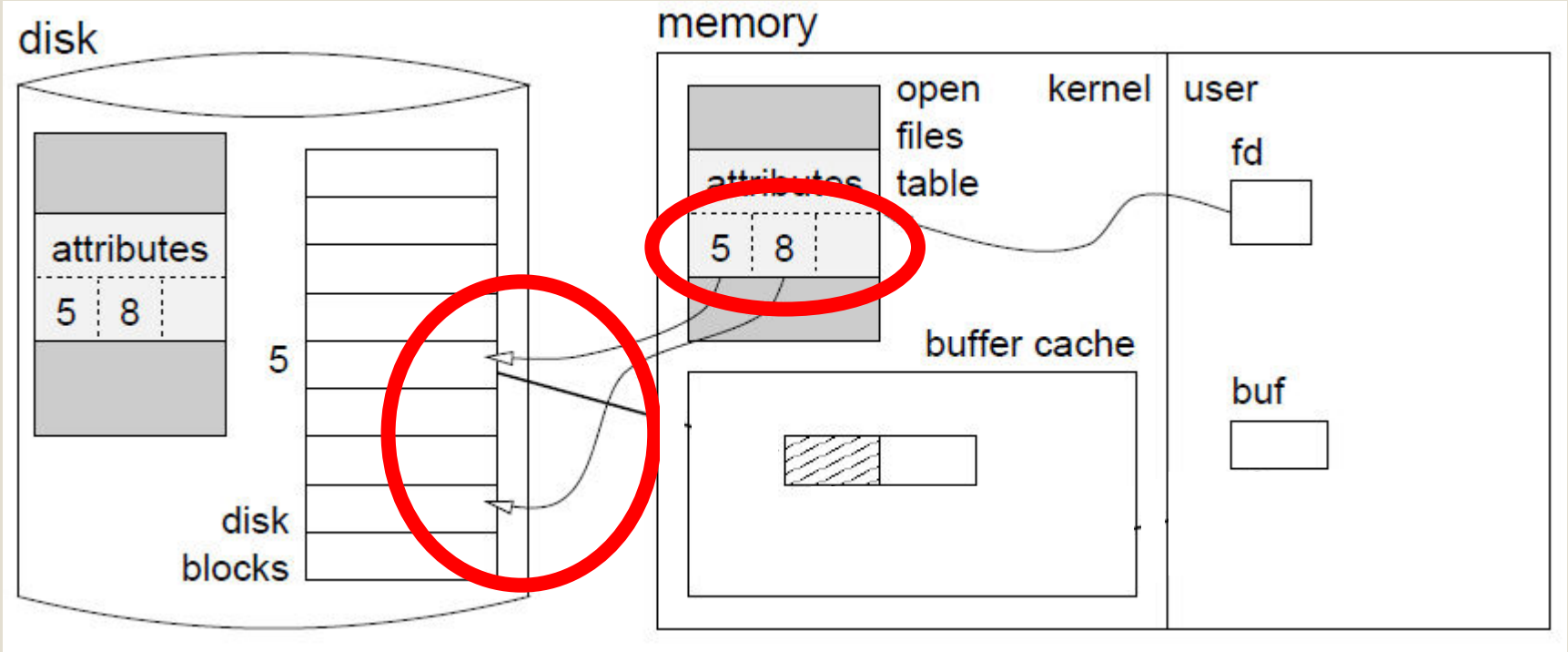
1. The argument fd identifies the open file by pointing into the kernel's open files table.



Reading from a File

read(fd,buf,100)

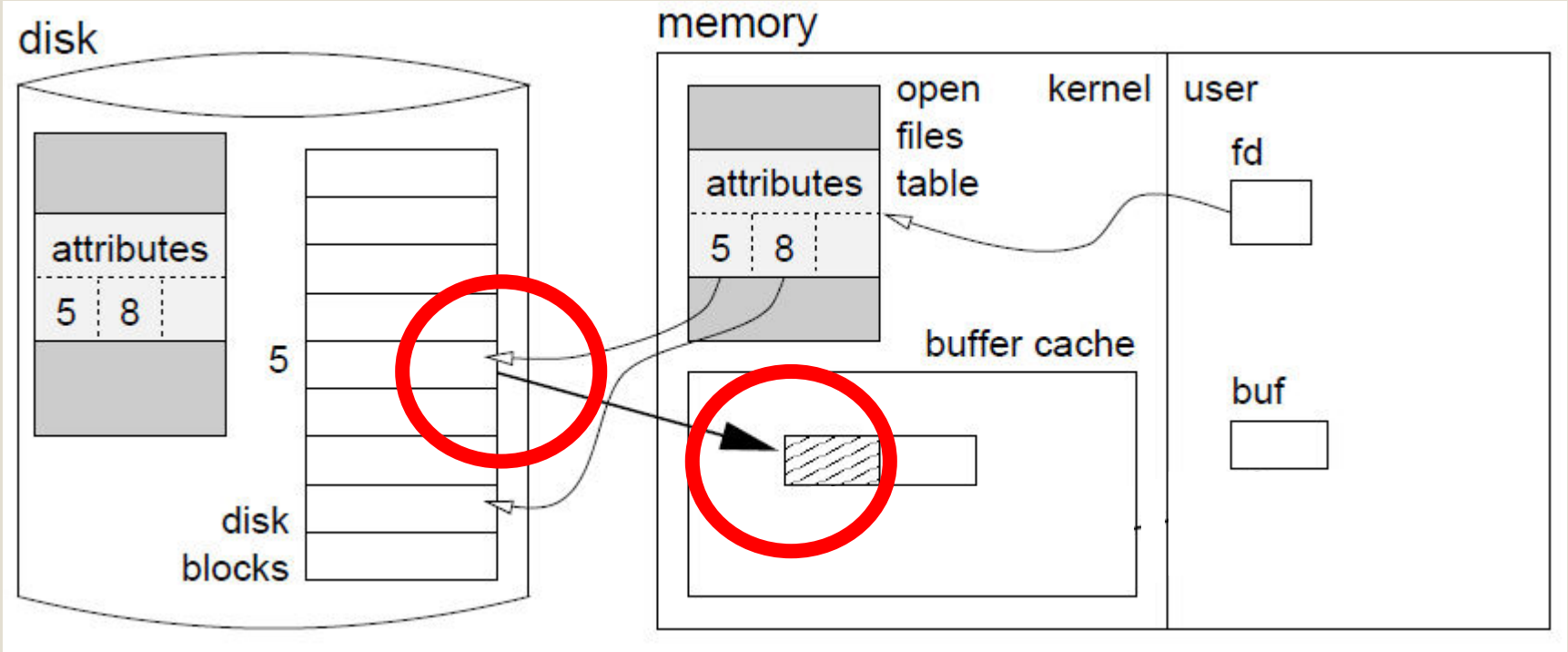
2. The system gains access to the list of blocks that contain the file's data.



Reading from a File

read(fd,buf,100)

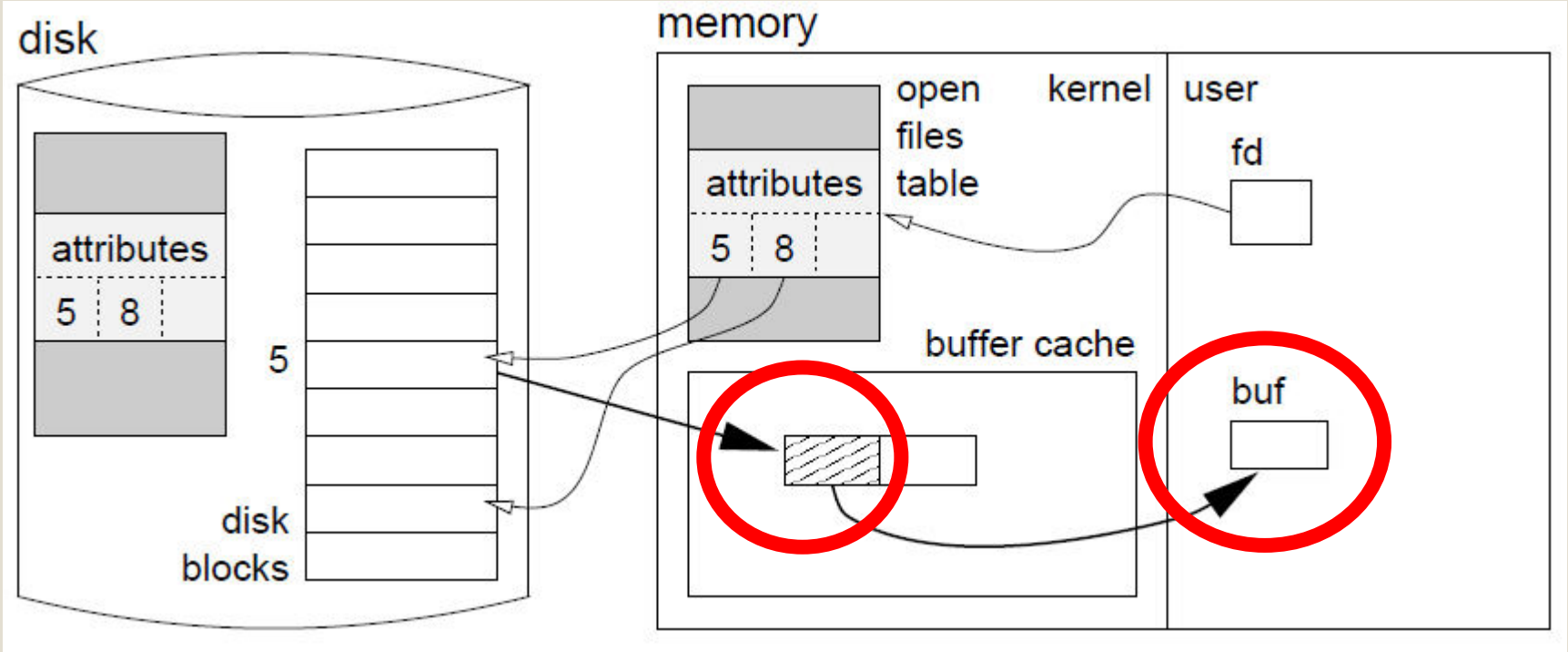
3. The file system reads disk block number 5 into its *buffer cache*. (full block = Spatial Locality)



Reading from a File

read(fd,buf,100)

4. 100 bytes are copied into the user's memory at the address indicated by buf.

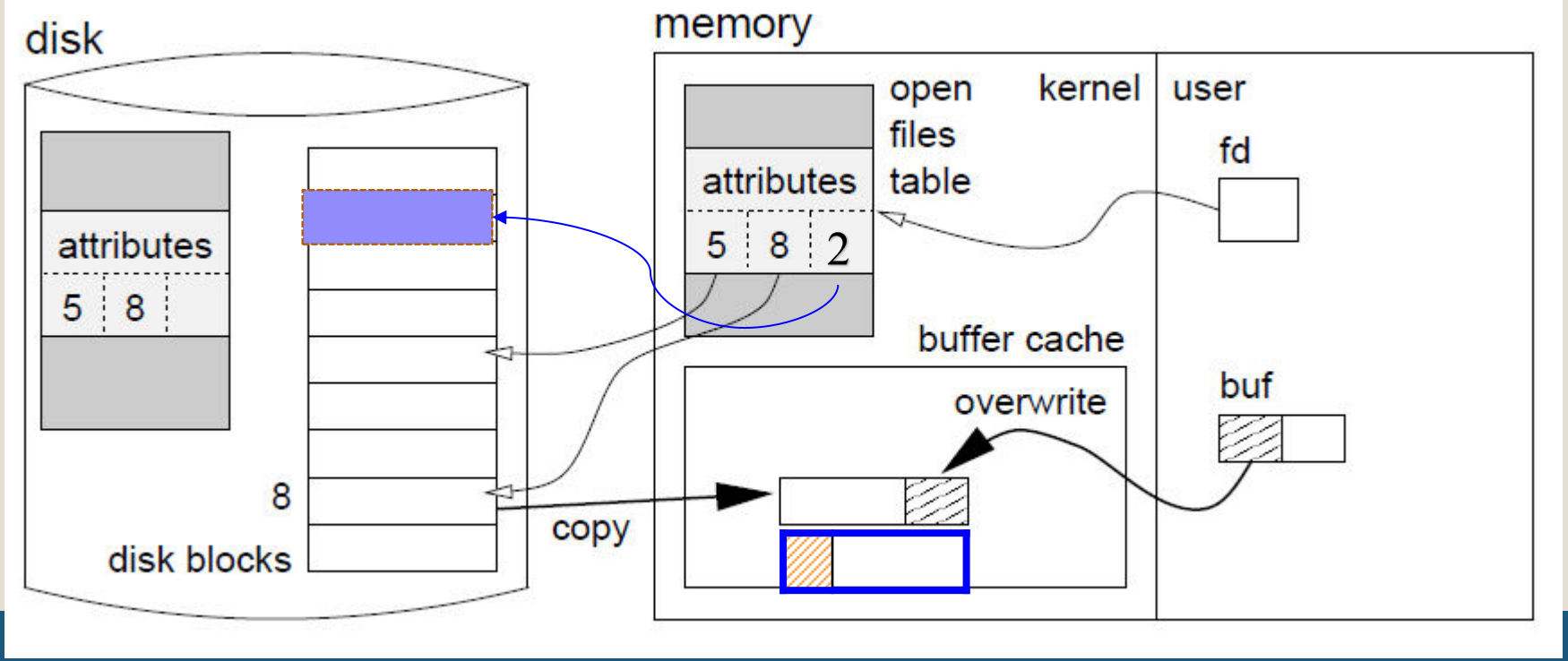


Writing to a File

- Assume the following scenario:
  - We want to write 100 bytes, starting with byte 2000 in the file.
  - Each disk block is 1024 bytes.
- Therefore, the data we want to write spans the end of the second block to the beginning of the third block.
- The full block must first be read into the buffer cache. Then the part being written is modified by overwriting it with the new data.

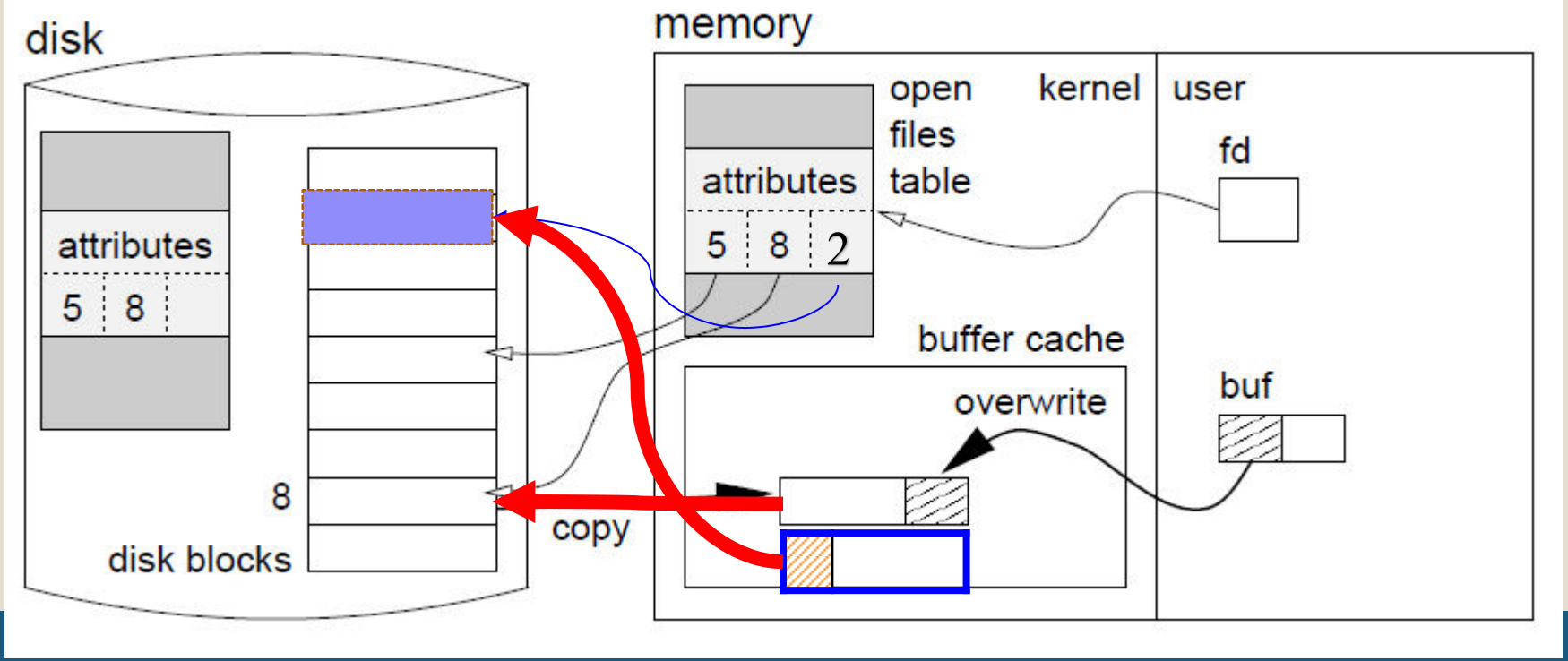
Writing to a File

- Write 48 bytes at the end of block number 8.
- The rest of the data should go into the third block.
- The third block is allocated from the pool of free blocks.



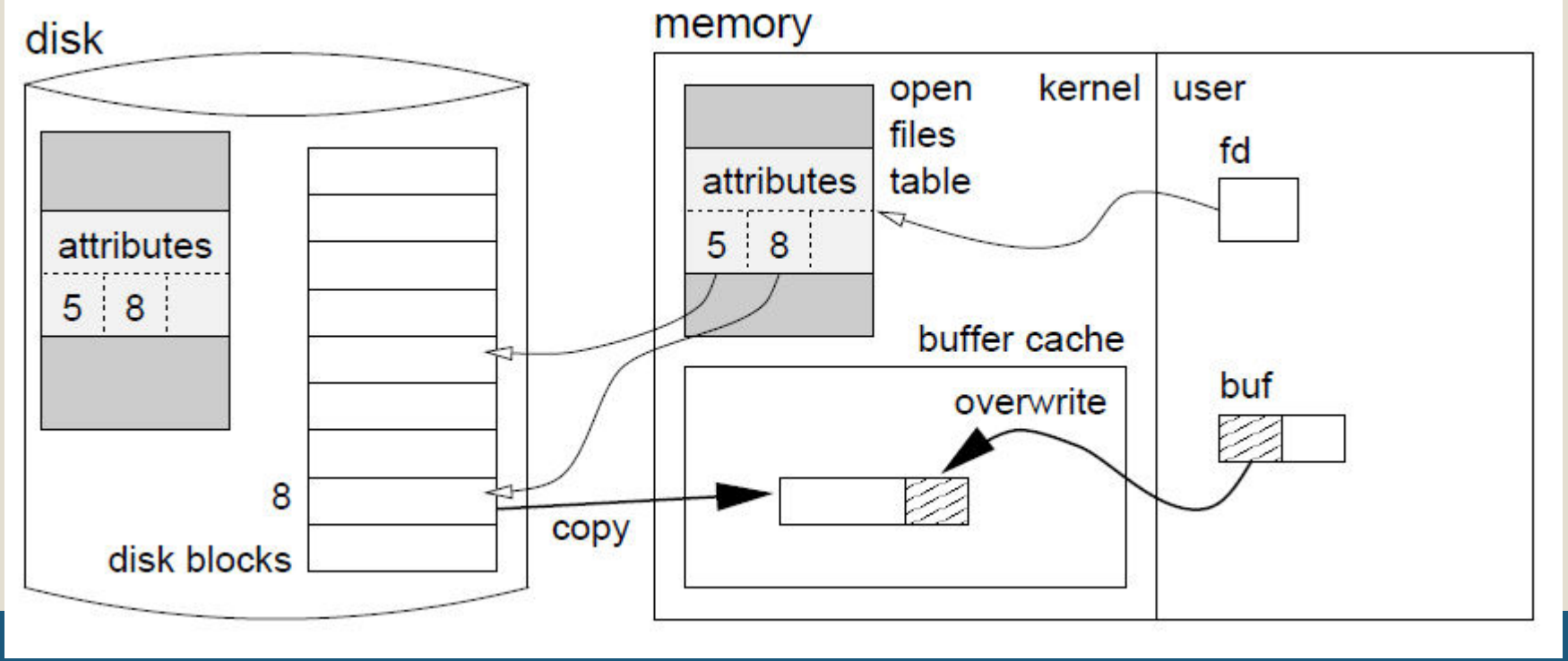
Writing to a File

- We do not need to read the new block from disk.
- Instead, we allocate a new block in the buffer cache
  - prescribe that it now represents block number 2.
  - copy the requested data to it (52 bytes).



Writing to a File

- Finally, the modified blocks are written back to the disk.



A note on the buffer cache

- The buffer cache is important to improve performance.

- But it can cause reliability issues:
  - It delays the writeback to disk.
  - Therefore if we are unlucky (shut down) the data may be lost.
  - Unplug USB.

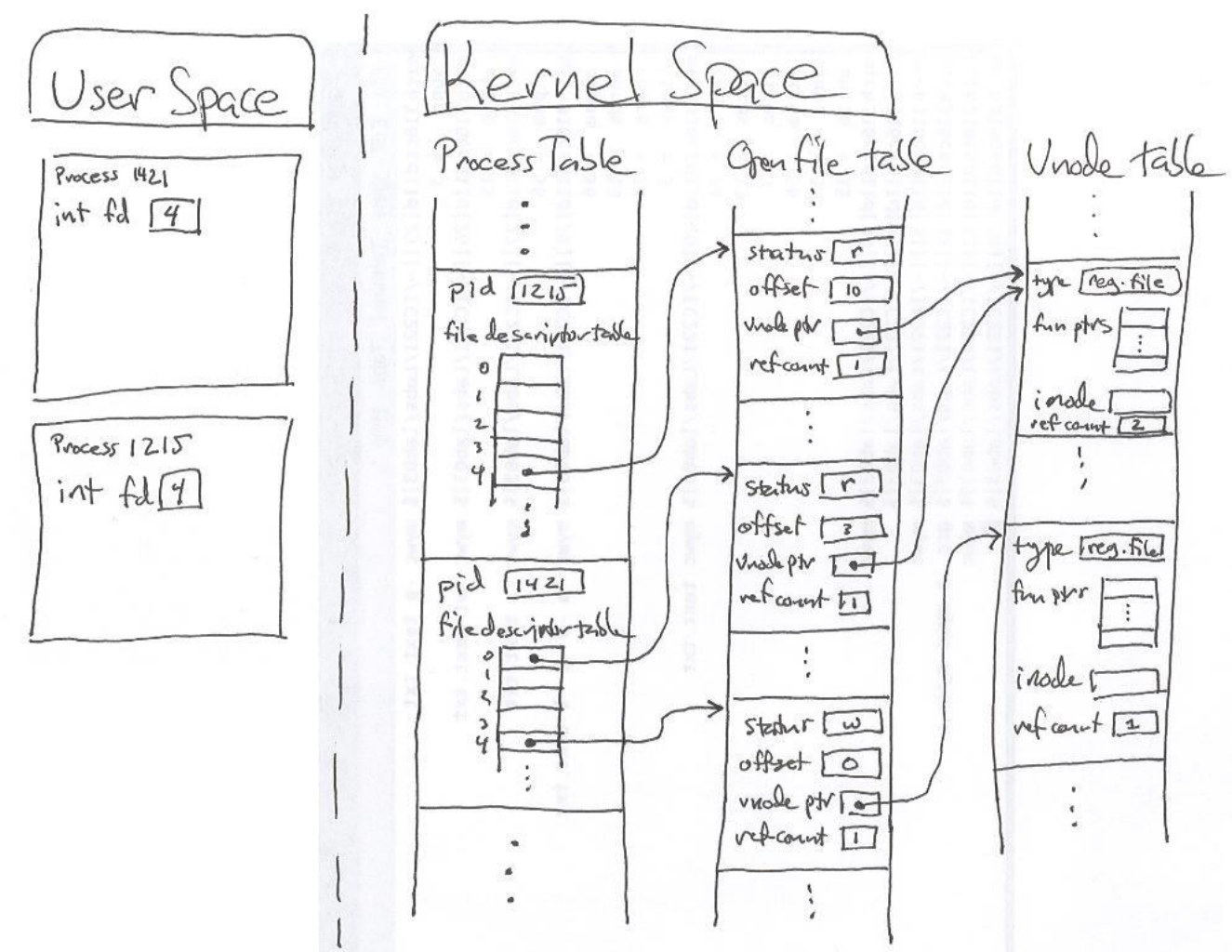
The Location in the File

- The **read** system-call is givven a buffer address for placing the data in the user's memory, without an indication of the **offset in the file** from which the data should be taken.
- The operating system maintains the **current offset** into the file, and updates after each operation.
- If random access is required, the process can set the file pointer to any desired value by using the **seek** system call.

The main OS file tables

- **The file descriptor table** – separate for each process. Each entry points to an entry in the open files table. The index of this slot is the fd that was returned by **open**.
- **The open files table** – an entry in this table is allocated every time a file is opened. Each entry contains a pointer to the inode table and a position within the file. There can be multiple open file entries pointing to the same i-node.
- **The i-node table** - each file may appear at most once in this table.

The main OS file table



Taken from: http://www.usna.edu/Users/cs/wcbrown/courses/IC221/classes/L09/Class.html

Questions from exams

מבחן 2012

6) כמה פעולות קריאה וכתיבה של בלוק מהדיסק צריך לבצע בהרצת התכנית הבאה (הנח שהקובץ קיים אבל ריק, גודל כל מדריך בלוק אחד,ושום דבר לא נמצא בזכרון מראש, ולא צריך לגשת לדיסק כדי להקצות בלוקים):

```
fd = open("/x/y/z/foo", O_CREATE);
write(fd, &buf, 13);
close(fd);
```

א. 5

ב. 8

ג. 11

ד. מספר אחר

Taken from: http://www4.huji.ac.il/exams/67808_2012_2_2_1.pdf

מבחן 2012

6) כמה פעולות קריאה וכתיבה של בלוק מהדיסק צריך לבצע בהרצת התכנית הבאה (הנח שהקובץ קיים אבל ריק, גודל כל מדריך בלוק אחד,ושום דבר לא נמצא בזכרון מראש, ולא צריך לגשת לדיסק כדי להקצות בלוקים):

```
fd = open("/x/y/z/foo", O_CREATE);
write(fd, &buf, 13);
close(fd);
```

א. 5

ב. 8

ג. 11

ד. מספר אחר

Taken from: http://www4.huji.ac.il/exams/67808_2012_2_2_1.pdf

טכניון 2005 מועד א'

שאלה 3: מערכות קבצים (24 נקודות)

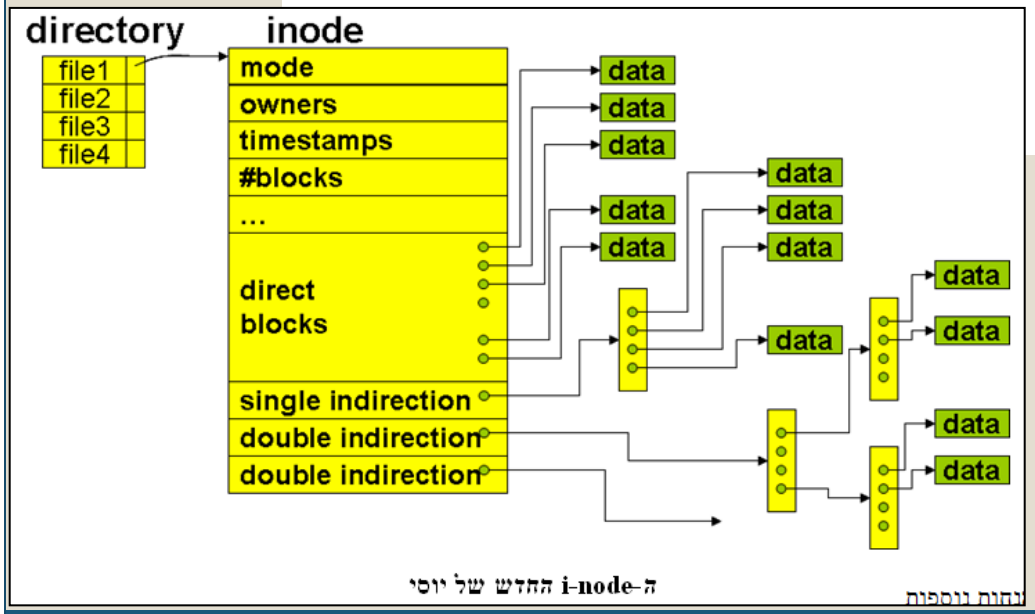
השאלה הבאה מתייחסת למבנה הנתונים i-node המשמש לאחסון נתונים על הדיסק, שנלמד בהרצאה על מימוש מערכות קבצים. (התרשים מההרצאה מופיע בעמוד הבא)

יוסי, הכותב את מערכת ההפעלה Yosix, ניגש לממש מערכת הקבצים. מאחר ונראה לו מסובך לממש Triple Indirection, הוא החליט להסתפק ב-double indirection שיבוא במקומו. בנוסף, מאחר והעריך כי רוב הקבצים במערכת יהיו קטנים, החליט להרבות במצביעים ישירים.

לפיכך המיפוי לבלוקי הקובץ מה-i-node שיצר נראה כך:

- 24 מצביעים ישירים לבלוקי הקובץ ממוספרים מ-direct 0 עד direct 23
- מצביע עקיף יחיד מסוג single indirection
- שני מצביעים מסוג double indirection

להלן תרשים של מבנה ה-i-node של יוסי:



ה-i-node החדש של יוסי

הנחות נוספות

הנחות נוספות

- גודל בלוק 512 bytes
- גודל מצביע 4 bytes

טכניון 2005 מועד א'

(4 נקודות)

א. ישנם נתונים רבים המופיעים ב-i-node הנ"ל, כגון mode, owners, timestamps, ... אולם הוא אינו כולל את ה-offset בקובץ. היכן נתון זה מוחזק?

_____

_____

טכניון 2005 מועד א'

(4 נקודות)

א. ישנם נתונים רבים המופיעים ב-i-node הנ"ל, כגון mode, owners, timestamps, ...
אולם הוא אינו כולל את ה-offset בקובץ. היכן נתון זה מוחזק?

נתון זה נמצא ב- file object שנמצא ב- Open Files Table

טכניון 2005 מועד א'

(3 נקודות)

ב. מהו הגודל המקסימלי של קובץ (בבלוקים) שניתן להצביע אליו באמצעות i-node זה, וכמה בלוקים סה"כ יתפוס קובץ זה על הדיסק (כולל index blocks , אולם ללא ה i-node של הקובץ)? יש להציג חישוב מפורט ולהסביר.

הגודל המקסימלי הוא ____ בלוקים. קובץ כזה יתפוס _____ בלוקים על הדיסק.

- Number of pointers in an indirect block: $\frac{512}{4} = 128$
- Number of data blocks: $24 + 128 + 2 \times (128)^2 = 32920$
- Total number of blocks: $32920 + 1 + 2 \times (128 + 1) = 33179$

טכניון 2005 מועד א'

(3 נקודות)

ב. מהו הגודל המקסימלי של קובץ (בבלוקים) שניתן להצביע אליו באמצעות i-node זה, וכמה בלוקים סה"כ יתפוס קובץ זה על הדיסק (כולל index blocks , אולם ללא ה i-node של הקובץ)? יש להציג חישוב מפורט ולהסביר.

הגודל המקסימלי הוא 32920 בלוקים. קובץ כזה יתפוס 33179 בלוקים על הדיסק.

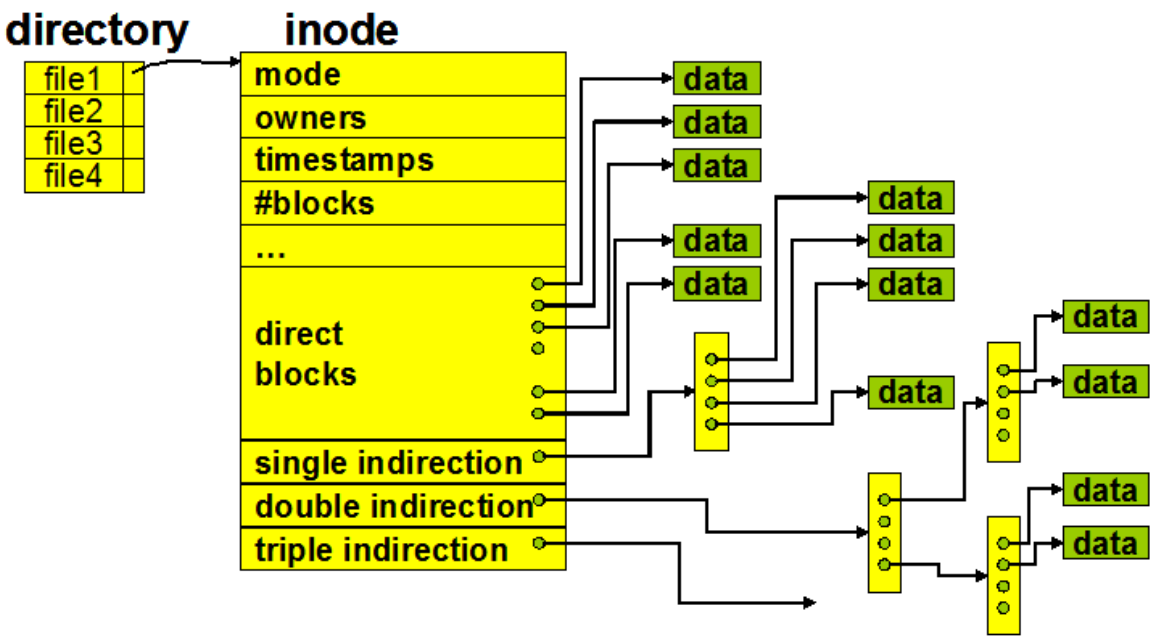
- Number of pointers in an indirect block: $\frac{512}{4} = 128$
- Number of data blocks: $24 + 128 + 2 \times (128)^2 = 32920$
- Total number of blocks: $32920 + 1 + 2 \times (128 + 1) = 33179$

טכניון 2005 מועד א'

ג. מהו הגודל המקסימלי של קובץ שניתן להצביע אליו לפי יצוג ה-i-node שנלמד בהרצאה? (בכל i-node יש 10 מחוונים ישירים, וכן מצביעים ל-3 רמות של indirection. ראו תרשים מצורף). כמה בלוקים סה"כ יתפוס קובץ זה על הדיסק (כולל index blocks , אולם ללא ה-i node של הקובץ)? יש להציג חישוב מפורט ולהסביר.

הגודל המקסימלי הוא _____ בלוקים. קובץ כזה יתפוס _____ בלוקים על הדיסק.

(3 נקודות)



מבנה i-node כפי שהוצג בהרצאה

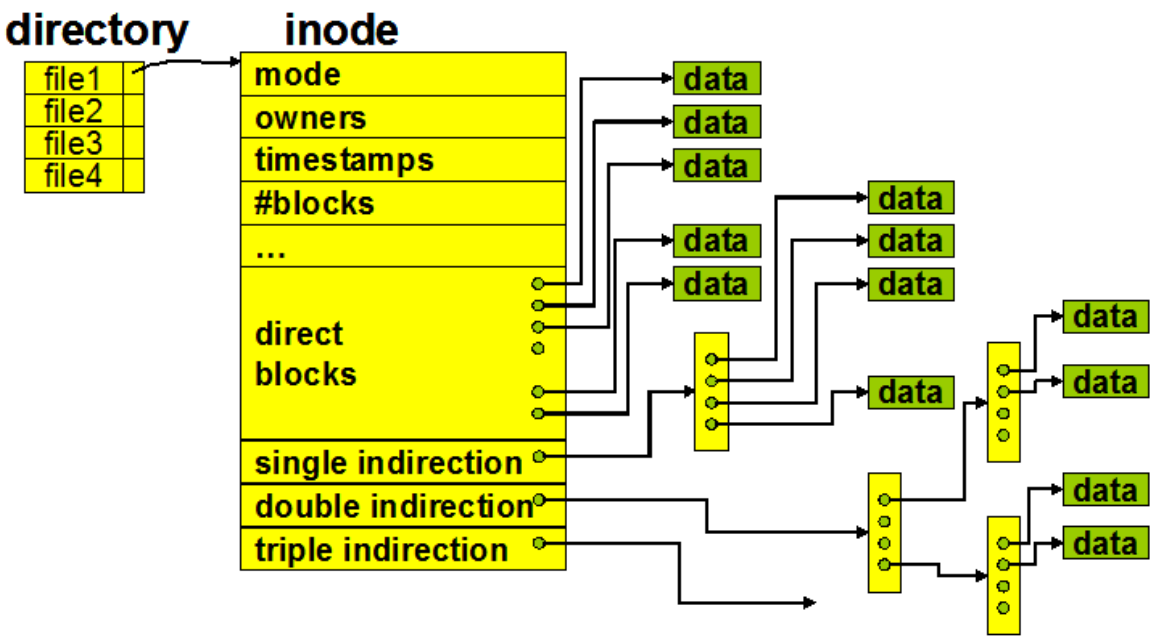
Taken from the Technion, 2005, Moed A

טכניון 2005 מועד א'

ג. מהו הגודל המקסימלי של קובץ שניתן להצביע אליו לפי יצוג ה-i-node שנלמד בהרצאה? (בכל i-node יש 10 מחוונים ישירים, וכן מצביעים ל-3 רמות של indirection. ראו תרשים מצורף). כמה בלוקים סה"כ יתפוס קובץ זה על הדיסק (כולל index blocks , אולם ללא ה-i node של הקובץ)? יש להציג חישוב מפורט ולהסביר.

הגודל המקסימלי הוא **2113674** בלוקים. קובץ כזה יתפוס **2130317** בלוקים על הדיסק.

(3 נקודות)



מבנה i-node כפי שהוצג בהרצאה

Taken from the Technion, 2005, Moed A

טכניון 2005 מועד א'

- Number of pointers in an indirect block: $\frac{512}{4} = 128$
- Number of data blocks: $10 + 128 + (128)^2 + (128)^3 = 2113674$
- Total number of blocks: $2113674 + 1 + (128 + 1) + (128^2 + 128 + 1) = 2130317$

טכניון 2005 מועד א'

בשאלות הבאות הניחו את ההנחות הבאות:

- השאלות מתייחסות לשיטת ניהול הקבצים הנהוגה ב-linux.
- בכל directory יש מספר קטן של קבצים.
- open() אינו מבצע prefetch (כלומר הקריאה של הנתונים עצמם תתבצע רק כאשר מבצעים read)

בספריה /usr/yossi ישנו קובץ בשם myfile שגודלו 32k.

ד. כמה בלוקים יתפוס הקובץ במערכת ההפעלה של יוסי? (כולל Index blocks, אולם ללא ה-i node של הקובץ?) (3 נקודות)

הקובץ יתפוס _____ בלוקים.

ה. כמה בלוקים יתפוס אותו קובץ לפי יצוג ה-i-node שראיתם בהרצאה? (לפי הנתונים מסעיף ג') (3 נקודות)

הקובץ יתפוס _____ בלוקים.

Taken from the Technion, 2005, Moed A

טכניון 2005 מועד א'

בשאלות הבאות הניחו את ההנחות הבאות:

- השאלות מתייחסות לשיטת ניהול הקבצים הנהוגה ב-linux.
- בכל directory יש מספר קטן של קבצים.
- open() אינו מבצע prefetch (כלומר הקריאה של הנתונים עצמם תתבצע רק כאשר מבצעים read)

בספריה usr/yossi/ ישנו קובץ בשם myfile שגודלו 32k.

ד. כמה בלוקים יתפוס הקובץ במערכת ההפעלה של יוסי? (כולל Index blocks, אולם ללא ה-i node של הקובץ?) (3 נקודות)

הקובץ יתפוס 65 בלוקים.

$$\text{Number of data blocks} = \frac{32KB}{512B} = 64 \rightarrow all\ directs + 40\ indirects \rightarrow \underbrace{64}_{data} + \underbrace{1}_{indirect1} = 65\ blocks\ in\ total$$

ה. כמה בלוקים יתפוס אותו קובץ לפי יצוג ה-i-node שראיתם בהרצאה? (לפי הנתונים מסעיף ג') (3 נקודות)

הקובץ יתפוס _____ בלוקים.

טכניון 2005 מועד א'

בשאלות הבאות הניחו את ההנחות הבאות:

- השאלות מתייחסות לשיטת ניהול הקבצים הנהוגה ב-linux.
- בכל directory יש מספר קטן של קבצים.
- open() אינו מבצע prefetch (כלומר הקריאה של הנתונים עצמם תתבצע רק כאשר מבצעים read)

בספריה usr/yossi/ ישנו קובץ בשם myfile שגודלו 32k.

ד. כמה בלוקים יתפוס הקובץ במערכת ההפעלה של יוסי? (כולל Index blocks, אולם ללא ה-i node של הקובץ?) (3 נקודות)

הקובץ יתפוס 65 בלוקים.

$$\text{Number of data blocks} = \frac{32\text{KB}}{512\text{B}} = 64 \rightarrow \textit{all directs} + 40\,\textit{indirects} \rightarrow \underbrace{64}_{data} + \underbrace{1}_{indirect1} = 65\,\textit{blocks in total}$$

ה. כמה בלוקים יתפוס אותו קובץ לפי יצוג ה-i-node שראיתם בהרצאה? (לפי הנתונים מסעיף ג') (3 נקודות)

הקובץ יתפוס 65 בלוקים.

$$\text{Number of data blocks} = \frac{32\text{KB}}{512\text{B}} = 64 \rightarrow \textit{all directs} + 54\,\textit{indirects} \rightarrow \underbrace{64}_{data} + \underbrace{1}_{indirect1} = 65\,\textit{blocks in total}$$

טכניון 2005 מועד א'

ו. נתונה קריאת המערכת הבאה:

```
open(fd, "/usr/yossi/myfile", O_RDONLY);
```

הניחו שהקובץ אליו ניגשים לקריאה קיים, והוא קובץ רגיל (לא soft/hard link).

מה המספר המינימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? ___ (2 נקודות)

נימוק:

_____

מה המספר המקסימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? ____ (2 נקודות)

נימוק:

_____

טכניון 2005 מועד א'

ו. נתונה קריאת המערכת הבאה:

open(fd, "/usr/yossi/myfile", O_RDONLY);

הניחו שהקובץ אליו ניגשים לקריאה קיים, והוא קובץ רגיל (לא soft/hard link).

מה המספר המינימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? **0** (2 נקודות)

נימוק:

כל הנתונים כבר בזיכרון הראשי (לא נדרשת גישה לדיסק)

מה המספר המקסימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? ____ (2 נקודות)

נימוק:

טכניון 2005 מועד א'

ו. נתונה קריאת המערכת הבאה:

```
open(fd, "/usr/yossi/myfile", O_RDONLY);
```

הניחו שהקובץ אליו ניגשים לקריאה קיים, והוא קובץ רגיל (לא soft/hard link).

מה המספר המינימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? **0** (2 נקודות)

נימוק:

כל הנתונים כבר בזיכרון הראשי (לא נדרשת גישה לדיסק)

מה המספר המקסימלי של גישות לדיסק שיתבצעו ע"י קריאה זו? **7** (2 נקודות)

נימוק:

מתבצעות גישות ל-/, /usr, /usr/yossi, שתי גישות לכל אחד מהם (פתיחה וקריאה), וגישה נוספת ל-/usr/yossi/myfile לפתיחה

מועד א 2020

במערכת קבצים מסוג i-node (כפי שראינו בכיתה), הניחו שכל i-node מכיל 6 מצביעים ישירים ועוד שלושה מצביעים עקיפים: single indirect, double indirect, triple indirect. שימו לב אנו מניחים שהשינוי היחיד במערכת הקבצים מזו שהוצגה בכיתה הוא במספר המצביעים.

גודל כל בלוק הוא 1024 בתים (1024 bytes), גודל כל מצביע לבלוק הוא 4 בתים (32 ביט), כל מדריך (תיקייה) מאוחסן בבלוק יחיד. הקובץ /a/b/c קיים ואורכו 10,000 בתים. כמו כן, הניחו שמיקומו של inode עבור "/" ידוע אך הוא אינו נמצא בזיכרון.

נניח שבתחילת כל פעולה (בכל אחד מהסעיפים הבאים) ה-disk cache ריק, גודלו אינסופי, וכל טבלאות הקבצים ריקות.

מועד א 2020

א. (2 נק') האם 36 בתים מספיקים לייצוג i-node במערכת כזו?

מועד א 2020

א. (2 נק') האם 36 בתים מספיקים לייצוג i-node במערכת כזו?

לא, נתון שגודל כל מצביע הוא 4 בתים, ושקיימים 9 מצביעים. לפיכך רק עבור המצביעים נדרשים 36 בתים. עם זאת, ב-, inodeכפי שנלמד בכיתה קיימים שדות נוספים, ולכן 36 בתים אינם מספיקים לייצוג ה-.inode

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ :/a/b/c

ידרשו _____ גישות בסה"כ, מתוכן:

_____ לבלוקים המכילים inodes בלבד,

_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

_____ לבלוקים המכילים את תוכן הקובץ.

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ :/a/b/c

ידרשו _____ גישות בסה"כ, מתוכן:

_____ לבלוקים המכילים inodes בלבד,

_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. 3 גישות למצביעים ישירים המבילים את תוכן הקובץ

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ /a/b/c:

ידרשו ___10___ גישות בסה"כ, מתוכן:

___ לבלוקים המכילים inodes בלבד,

___ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

___ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. 3 גישות למצביעים ישירים המבילים את תוכן הקובץ

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ :/a/b/c

ידרשו ___10___ גישות בסה"כ, מתוכן:

___4___ לבלוקים המכילים inodes בלבד,

_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. 3 גישות למצביעים ישירים המבילים את תוכן הקובץ

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ /a/b/c:

ידרשו ___10___ גישות בסה"כ, מתוכן:

___4___ לבלוקים המכילים inodes בלבד,

___0___ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. 3 גישות למצביעים ישירים המבילים את תוכן הקובץ

מועד א 2020

ב. (4 נק') חשבו כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 2000-3700 מהקובץ :/a/b/c

ידרשו ___10___ גישות בסה"כ, מתוכן:

___4___ לבלוקים המכילים inodes בלבד,

___0___ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

___3___ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. 3 גישות למצביעים ישירים המבילים את תוכן הקובץ

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?
ידרשו _____ גישות בסה"כ, מתוכן:
_____ לבלוקים המכילים inodes בלבד,
_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),
_____ לבלוקים המכילים את תוכן הקובץ.

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?
ידרשו _____ גישות בסה"כ, מתוכן:
_____ לבלוקים המכילים inodes בלבד,
_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),
_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. גישה לבלוק מצביעים דרך מצביע מסוג SINGLE INDIRECT ולבלוק של תוכן הקובץ המוצבע ממנו

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?
ידרשו _____9_____ גישות בסה"כ, מתוכן:
_____ לבלוקים המכילים inodes בלבד,
_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),
_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. גישה לבלוק מצביעים דרך מצביע מסוג SINGLE INDIRECT ולבלוק של תוכן הקובץ המוצבע ממנו

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?
ידרשו ___9___ גישות בסה"כ, מתוכן:
___4___ לבלוקים המכילים inodes בלבד,
_____ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),
_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. גישה לבלוק מצביעים דרך מצביע מסוג SINGLE INDIRECT ולבלוק של תוכן הקובץ המוצבע ממנו

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?

ידרשו ___9___ גישות בסה"כ, מתוכן:

___4___ לבלוקים המכילים inodes בלבד,

___1___ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

_____ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. גישה לבלוק מצביעים דרך מצביע מסוג SINGLE INDIRECT ולבלוק של תוכן הקובץ המוצבע ממנו

מועד א 2020

ג. (4 נק') חשבו שוב כמה גישות לדיסק נצטרך על מנת לקרוא את הבתים 7200-8000 בקובץ זה?

ידרשו ___9___ גישות בסה"כ, מתוכן:

___4___ לבלוקים המכילים inodes בלבד,

___1___ לבלוקים המכילים מצביעים בלבד (עבור הצבעות עקיפות single\double\triple indirect),

___1___ לבלוקים המכילים את תוכן הקובץ.

הגישות הנדרשות הן:

1. ל INODE של \ ול DATA של \
2. ל INODE של a ול DATA של a
3. ל INODE של b ול DATA של b
4. ל INODE של c ול DATA של c
5. גישה לבלוק מצביעים דרך מצביע מסוג SINGLE INDIRECT ולבלוק של תוכן הקובץ המוצבע ממנו

Q & A