CSCB09: Software Tools and Systems Programming

Bianca Schroeder
bianca@cs.toronto.edu
IC 460

Who am I

- Professor with research area in computer systems / system reliability
- Lots of research collaborations with companies (e.g. Google, Network Appliances) and supercomputing centers.
- Associate department chair for computer science
- Teaching: B09 or C69, grad courses on reliability and storage systems.

Administrivia - Communications

- Main constraint:
  - There is 200 of you, but only one of me …
- Therefore:
  - Step 1: Talk to your TA in the tutorial
  - Step 2: Post on Piazza
    - Piazza (you should have received an invitation)
    - Make sure to read their "Terms of Use" and contact me if you are not comfortable
  - Only if step 1 & 2 fail: email me
    - Email: bianca@cs.toronto.edu
      - Email must include your name.
      - Subject must include B09
    - Email is a formal method of communication:
      - State your question clearly, with enough context.
      - Sign it (Name and utorid are the most useful.)

Course Information

- Check the course information sheet (on the course web page)
- The course web page and Piazza are the official sources of announcements.

http://www.cs.toronto.edu/~bianca/cscb09s17/

https://piazza.com/class/j29k4eb91xu7jy

- It's your responsibility to follow Piazza (e.g. find out about changes to assignment deadlines & requirements)

Course work

- A1: Shell and very basic C programming
- A2: Pointers, memory management (C)
- A3: Fork and pipes (C)

- PCRS work
- One midterm, one final.
- Final: Must receive >= 40% to pass course

Assignments

- A1: Shell and very basic C programming
- A2: Pointers, memory management (C)
- A3: Fork and pipes (C)

•Assignments:

- Submission through SVN and Markus
- All code **must** work on the lab machines (BV473/ mathlab) to receive full marks.
  - Code that does not compile on lab machines will receive 0.
  - Code that does not run will not get any marks from automarking
  - *Don't wait until the last day!*

Did you catch that?

Code that does not compile will receive a grade of 0

(Almost) Weekly Tutorials

- Labs
  - Starting next week, in BV 473
  - Work in pairs
  - To prepare you for the assignments

The inverted classroom

- Preparation before class (videos+exercises) on PCRS:
  https://cms-pcrs.utsc.utoronto.ca/cscb09/content/quests
- Hands-on activities in class
  - Need to bring computers to class (or share)

The inverted classroom

- For next week work on “Week 2 preparation” on PCRS:
  https://cms-pcrs.utsc.utoronto.ca/cscb09/content/quests

- Complete all modules that are marked as “Credit”, i.e.:
  - “Input, Output and Compiling”, “Iteration”, “Arrays”.

Plagiarism

- “The work you submit must be your own, done without participation by others. It is an academic offense to hand in anything written by someone else without acknowledgement.”
- You are not helping your friend when you *give* him or her a copy of your assignment.
- You are hurting your friend when you *ask* him or her to give you a copy of their assignment.

What is cheating?

- Cheating is
  - copying parts or all of another student's assignment
  - including code from books, web sites, other courses without attribution
  - getting someone else to do substantial parts of your assignment
  - giving someone else your solution
- Cheating is not
  - helping to find a bug in a friend's code (be careful)
  - helping each other understand man pages or example code.
- We will run a plagiarism detector on each assignment!

Course Overview

- Unix Software Tools
  - Understanding the shell
  - Shell programming
  - Make
- Systems Programming
  - C
  - files
  - processes
  - concurrency
  - communication

Windows users

- You can install **cygwin**, which gives you a linux-like environment on windows.
- All code must run on the lab machines, so if you develop C code elsewhere make sure it works on the lab machines (or mathlab).
- If you want to work remotely on the lab machines or mathlab, you can **install putty** on windows, which allows you to connect remotely. On linux machines you can use **ssh.**

- As an instructor I cannot provide tech-support for any of the above. However, you may ask Tianze in IC464 or your TA.

Self Study Topics

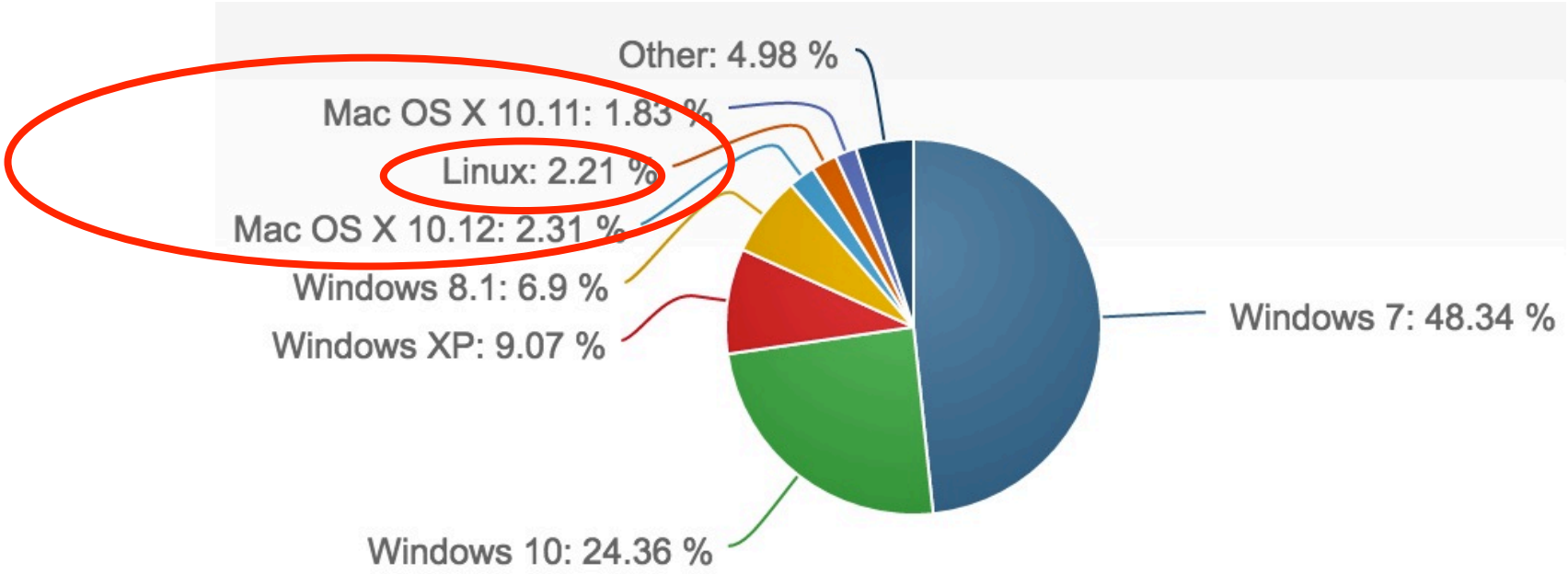
- Using SVN
- Using basic Unix
- Learning an editor – vi, emacs, scite, nedit, …

- We will spend a bit of time on these in class/tutorial, but not a lot.

The plan for today

- What is Unix and why are we using it?
- What is an operating system?
- What are files and directories?
- What is a shell and why do I care about shell programming?
- Along the way: learn/refresh basic Unix commands

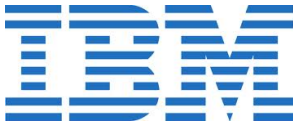
Is anybody using Unix/Linux?



- %age of the world's laptops/desktops running Unix is smallish …

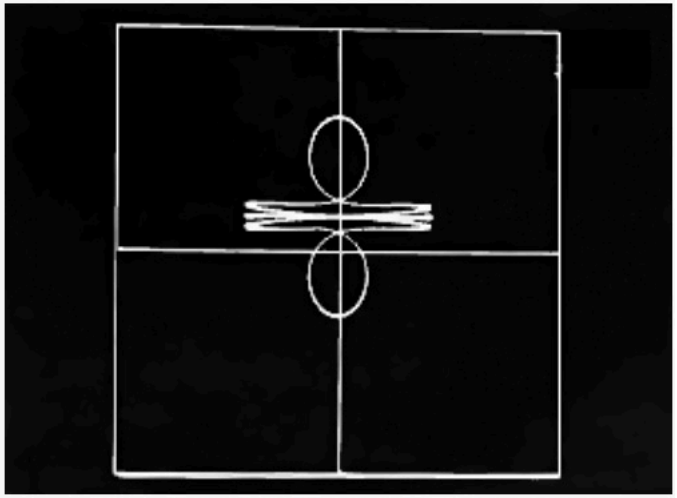
Is anybody using Unix/Linux?

- %age of laptops/desktops running Unix?
  - 8%
- %age of tablets/smartphones running Unix?
  - 95%
- %age of web servers running Unix?
  - 65%
- %age of supercomputers running Unix?
  - 98%



Why Unix?

- 1969:
  - Ken Thompson at Bell Labs wanted to play Space Travel on his DEC PDP-7
  - Wrote the first version in assembler in one month



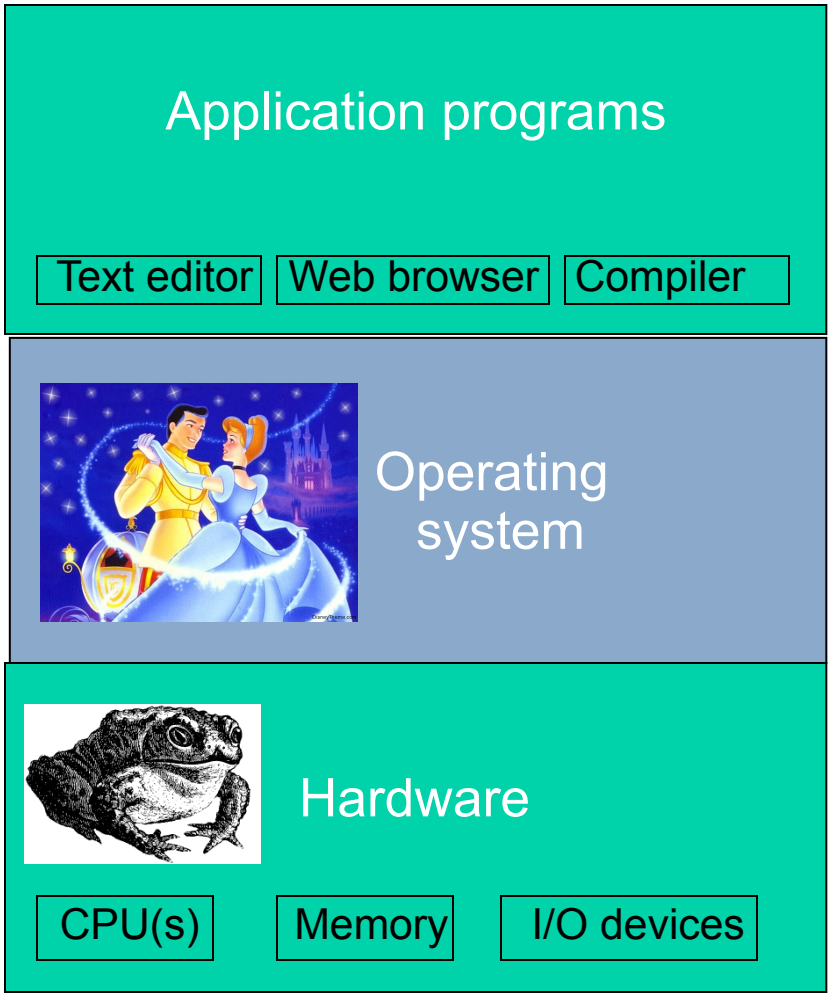
Gameplay image of *Space Travel*.

- Now:
  - Available on many platforms
  - Multi-user, multi-programmed
  - Good at
    - sharing computer resources
    - manipulation of files, processes, programs
    - inter-process and inter-machine communication

Why Unix?

- Open source
  - Often free
  - Access to operating features
  - Can be customized

What is an operating system?

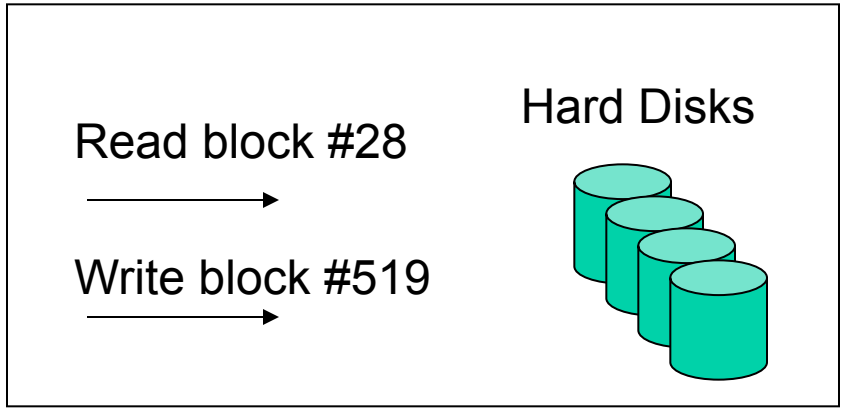


The software layer between user applications and hardware.

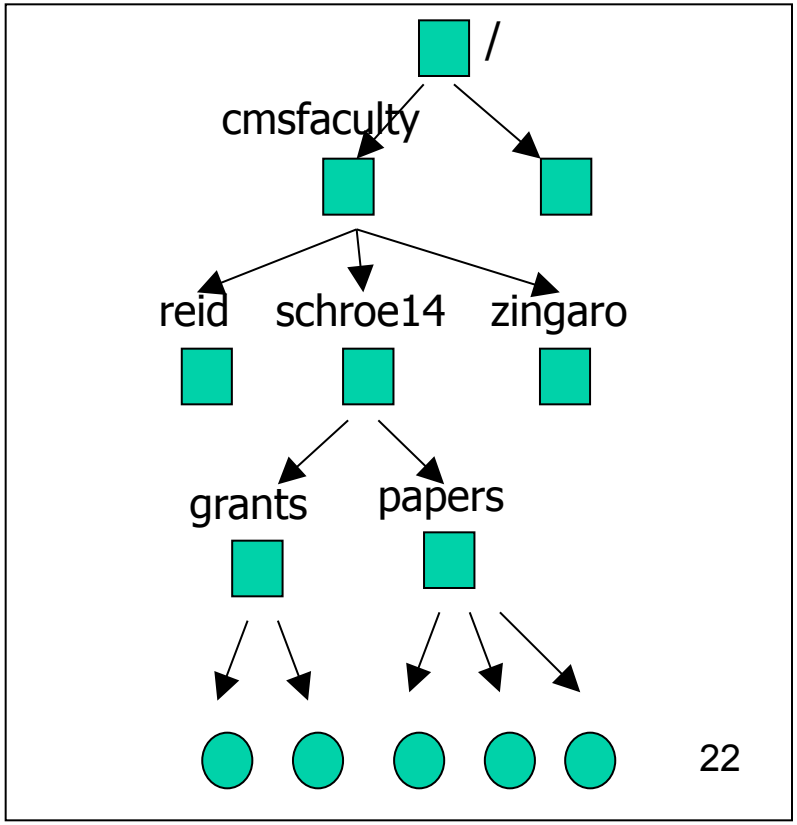
- Serves as a resource manager
  - allows proper use of resources (hardware, software, data)
- Serves as a control program (protection)
  - controls execution of user programs to prevent errors and improper use of the computer
- Turns ugly hardware into beautiful abstractions (provides services)

One such abstraction: Files and Directories

Reality



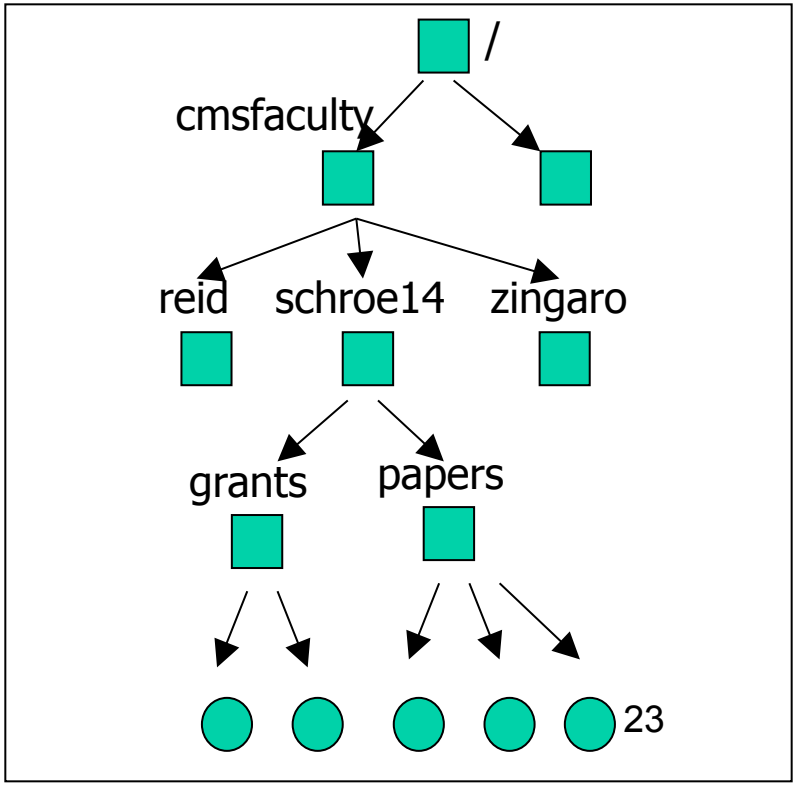
Abstraction



Unix commands to navigate directories and files

- Unix has the notion of a *current working directory*. Get it by typing `pwd` in the shell.
- Type `ls` to see all files and directories in current working directory.
- Use `cd` to change your working directory.
- Use `cat` or `more` to view file contents.
- Use `cp`, `mv`, `rm` to copy / move/ remove files.
- Use `mkdir` to create a directory.

- (Use man pages to learn more about commands, e.g. type "`man ls`" for info on the ls command)

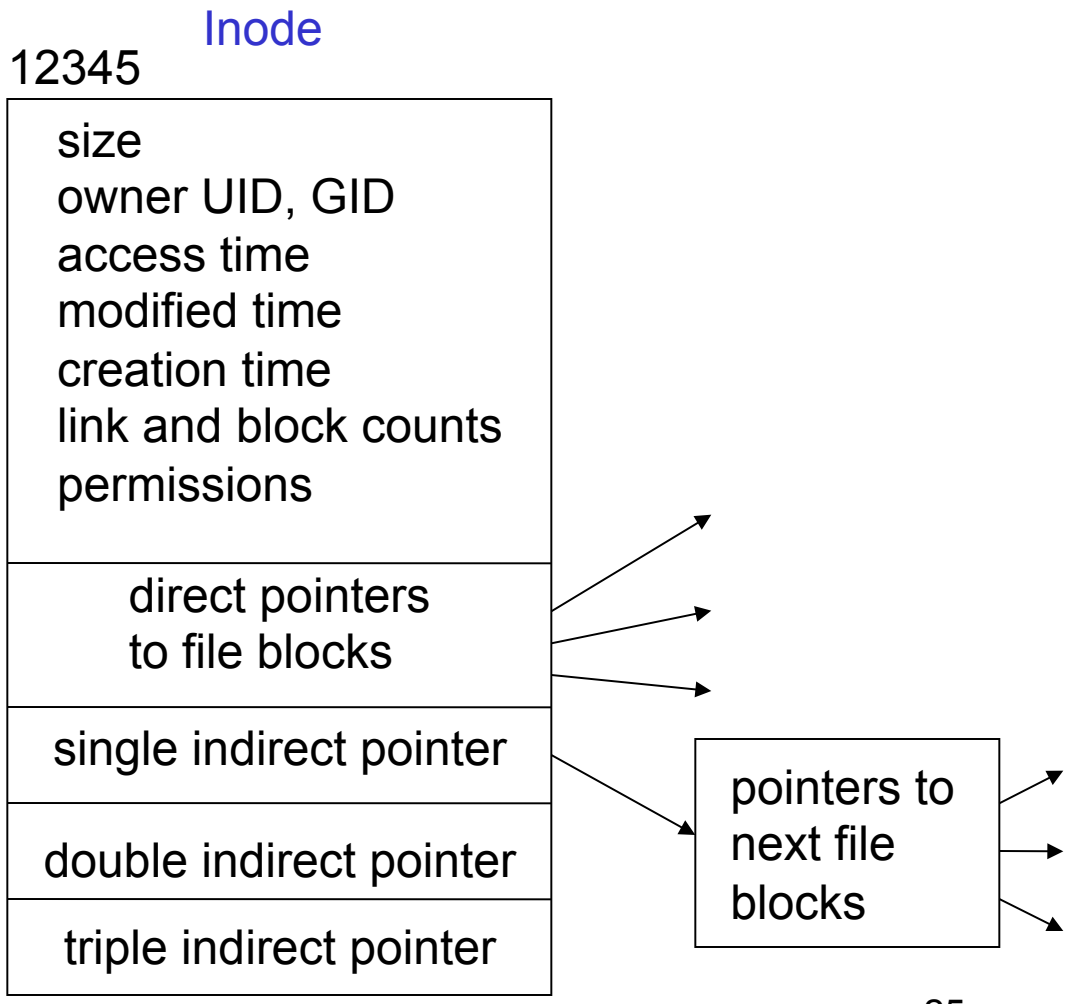


What is a file?

- A named collection of data with some attributes
  - Name
  - Owner (user and group)
  - Size
  - Permissions
  - Time of creation, last access, last modification
  - Location on disk
- How to get information on a file in Unix?
  - `ls -l, stat`

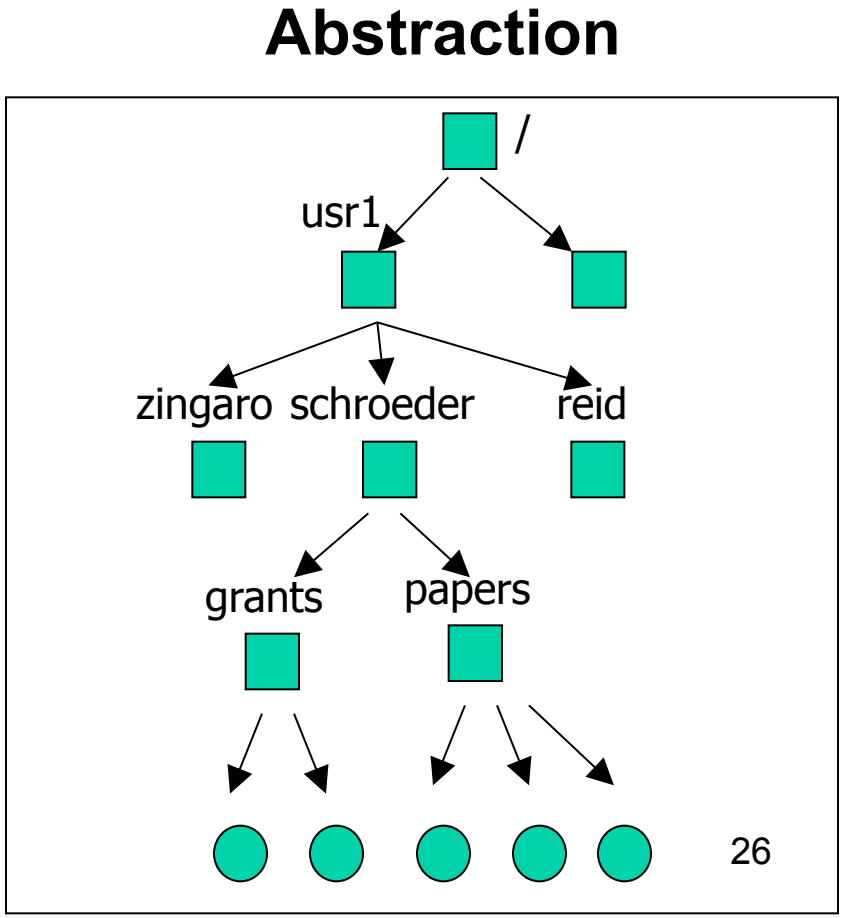
How does Unix implement a file?

- A data structure called **inode** stores the information about a file, including which disk blocks contain the file data.
- Identified by its **inode number**



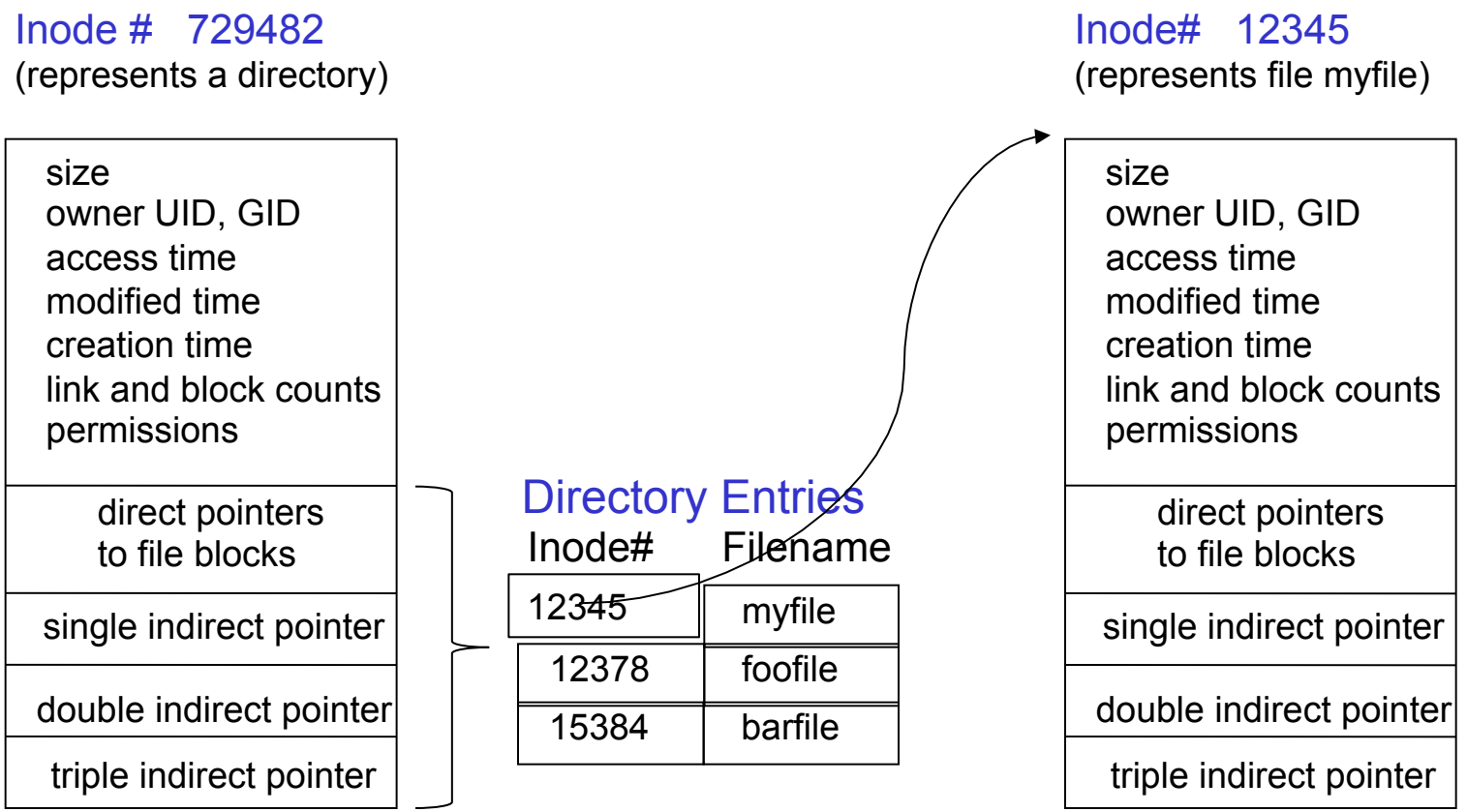
What is a directory?

- A collection of files (and sub-directories)
- One special directory: the root directory (named “/”)
- In Unix: internally every directory is implemented as a file!



What is a directory?

- Internally, every directory is a file itself!
  - File with special content: *directory entries*
  - A directory entry maps a file name to an inode.



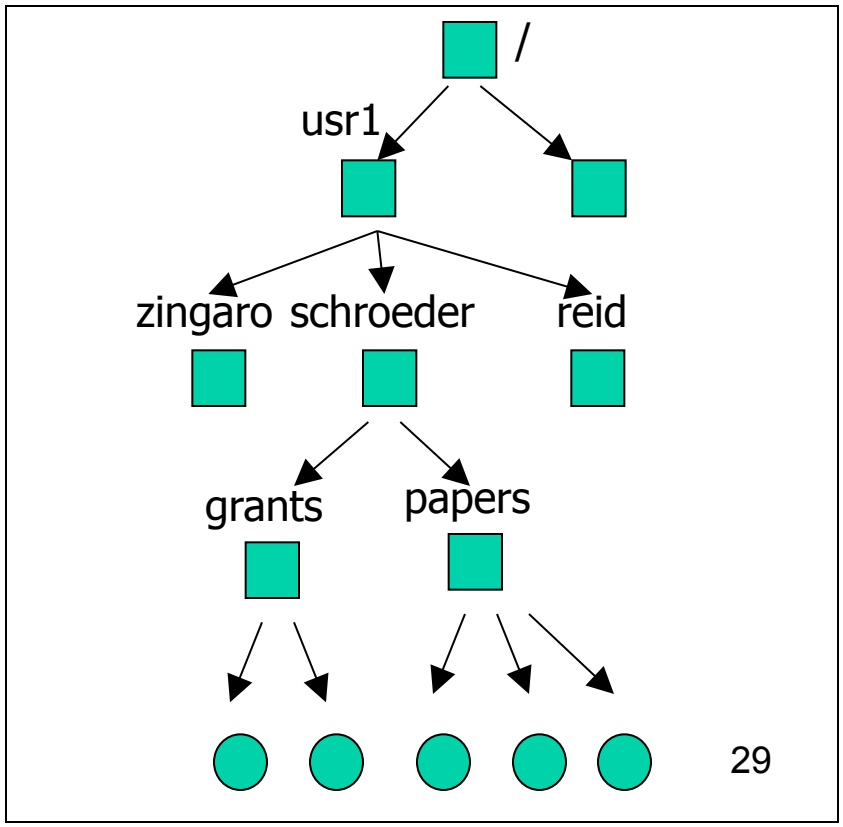
Many more things are files …

- "Everything is a file."
- Unix provides a file interface for all Input/Output.
  - regular files
  - directories
  - devices
    - video (block)
    - keyboard (character)
    - sound (audio)
    - network (block)
- File interface = open, read, write, close

The directory hierarchy

- The directory tree is not really a tree as shown in the previous pictures…
  - But an acyclic graph
  - Why?

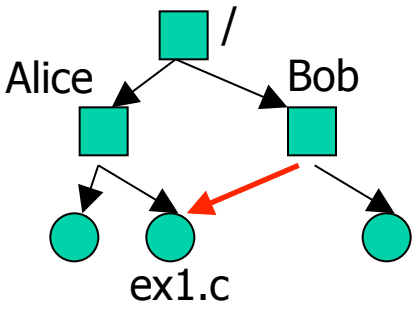
Abstraction



The directory hierarchy

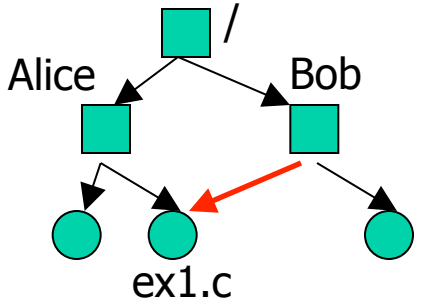
- Sharing of files can be implemented by creating a new directory entry called a *link* : a pointer to another file or subdirectory

- An `"ls"` in Bob's home directory
  will show ex1.c



Hard Links

- Create with "`ln <target> <name of link>`"
- Second directory entry identical to the first
- What happens if anybody removes (`rm`) ex1.c?
  - A file is only removed if it no longer has any name/ hard links.



'~Alice' directory

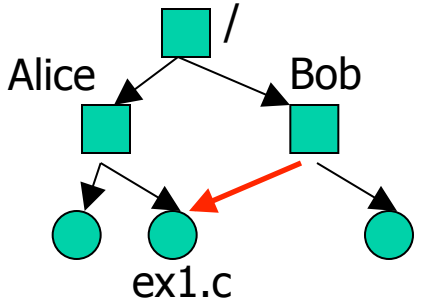
File Name	Inode	Type
…	…	…
ex1.c	42	file
…	…	…

'~Bob' directory (hard link)

File Name	Inode	Type
…	…	…
ex1.c	42	file
…	…	…

Soft (symbolic) Links

- Create with "`ln -s <target> <name of link>`"
- Directory entry points to small file containing the true path of the linked file
- What happens if anybody removes (`rm`) ex1.c?
  - If Alice removes ex1.c, dangling reference
  - If Bob removes ex1.c, only the entry in his directory is removed



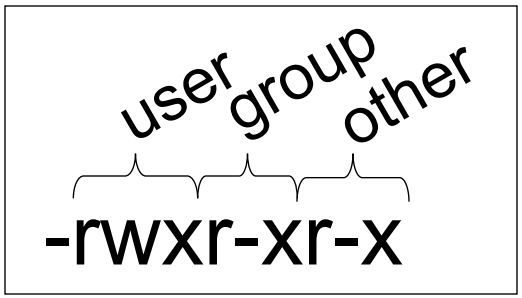
'~Alice' directory

File Name	Inode	Type
…	…	…
ex1.c	42	file
…	…	…

'~Bob' directory (soft link)

File Name	Inode	Type
…	…	…
ex1.c	529	link
…	…	…

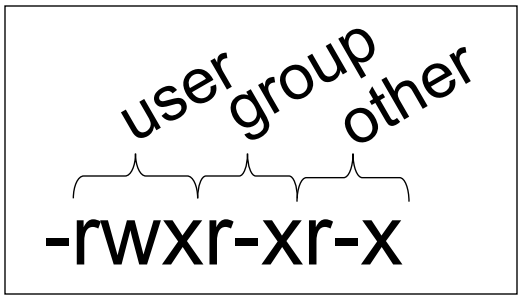
"/Alice/ex1.c"



File Permissions

```
-rwxr-xr-x 1 schroe14  cmsusers    0 Jan  6 11:35 file1
-r--r--r-- 1 schroe14  cmsusers    0 Jan  6 11:35 file2
-rw------- 1 schroe14  cmsusers    0 Jan  6 11:35 file3
```

- Each file is owned by a particular user and group
- Each file has three permission entries
  - Permissions for owning user
  - Permissions for owning group
  - Permissions for all other users
- Each entries specify three permissions:
  - read, write, execute – pretty much what you think



Directory Permissions

```
drwxr-xr-x 2 schroe14  cmsusers 512 Jan  6 11:36 dir1/
dr-x--x--x 2 schroe14  cmsusers 512 Jan  6 11:36 dir2/
dr--r--r-- 2 schroe14  cmsusers 512 Jan  6 11:36 dir3/
```

- Directory permissions
  - read – you can run `ls` on the directory
  - write – you can create and delete files in the directory
  - execute – you can "pass through" the directory when searching subdirectories (you can `cd` into it) and access files (according to files' permissions).

How can you change permissions?

- `chmod`
- Lots of ways, check `man chmod`
- E.g. give user owning file `fname` execute permissions:

```
chmod u+x fname
```

- Give all users in group read permissions

```
chmod g+r fname
```

- Remove read permissions from all users in group

```
chmod g-r fname
```

- Give all users all permissions

```
chmod a+rwx fname
```

That’s it for today!