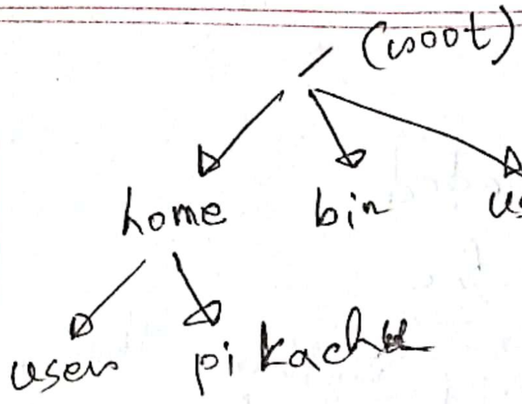
CSE 314

07/06/2023



Default Variables –
$USER, $PATH

cd ~ → /home/user (currently logged-in user)
cd / → /

cd /home/... → absolute path, always starts from root

cd a/b/... → relative path, w.r.t pwd

cd .. → PARENT folder, not back

cd ../c → Go to parent, then 'c' folder in them

ls -l → detailed info of the files (-l: switch)

.filename → hidden file

Folder size → Folder এর ভিতরের files রাখার জন্য কিছু info রাখতে হয় তার size। ভিতরের content এর size না।

mkdir 1905{001..120} → create directories named 1905001 - 1905120

mkdir 1905{001..120}-{A..Z} ~~→ 1905~~

mkdir (-p) submission/....

↓

create directory as needed

cp (-r) submission copy-loc

↓ ↓ ↓

copy all files & subdirectories recursively

src directory

copy within this folder

rm (-r) copy-loc

↓ ↓

recursively remove all files & subdirectories

directory to remove

rm (-f) filename.ext

↓

delete if present, ignore otherwise, thus don't show errors.

rm -r 1905{001..010}

mv submission/code.c copy-check/code3.c

↓ ↓

src dest

mv doutrename.txt renameme.txt

↓ ↓

file to be renamed

final filename

rm *.txt → delete all text files
rm *code.c → all files with code.c prefix deleted.
rm * → Current folder-এ যা আছে সব delete করার চেষ্টা করবে, file delete হবে, folder delete করতে -ও লাগবে।

cd — → toggle between pwd and immediate previous directory.

pushd /bin/X11 → push current folder onto a stack and "cd" to the folder "/bin/X11"

Recursive tasks

popd → "cd" to the top-of-the-stack
dirs → to see the stack, although it always shows pwd no matter whether stack exists/not.

Entity

User Owner (u)
Group (Linux-এ কয়েকজনের owner/user মিলে group করা যায়) (g)
others (বাকিরা যারা owner) (o)

Permission

Read (r)
Write (w)
Execute (x)

zsh

file execute → .out file-ও read excess থাকলে read করা যাবে, "x" permission না থাকলে run করা যাবে না।

chmod e (±) p filename.ext [give/revoke]
e → entity
p → permission

superuser না হলে "sudo" command use করা যাবে না। Linux install করার সময় যে user → Superuser.

whoami → shows currently logged-in user.

vim, vi, nano → edit files by opening in terminal

sudo userdel -r pikachu → Delete all files folders created in "/home/pikachu" as well as "pikachu"

head -n 5 file.txt → show first 5 lines of file.txt

ls directory | sort
ls directory → Output of this cmd
| → =
sort → Input of this cmd

CSE 319 19/06/2023

Trap → System call, fault

Trap vs Interrupt [Timer, I/O]

ecall → Mode switch → external

Fault → যখন Memory-র ভেতর ঢোকা হয়.

→ OS বিভিন্ন trap করতে পারে

→ User → Machine → Kernel

usys.S, usys.pl, trap.c, syscall.c, trampoline.S,
syscall.h, user.h

auto-generated/
auto-modified

sysfile.c
sysproc.c

file-এর handler
করে হয়, Memory
issues, এই আলাদা
করে রাখা

ao registers
↳ stores return values

proc.c, defs.h

add a new function that I want to call from syscall.c

And add declaration of this new function here

→ normally process-related functions
থাকে

OS-এ কোনো variable থাকলে global হয়।

☐| Heuristic Functions:

- The cost of an optimal solution to a "relaxed problem" is an admissible heuristic for the original problem.
  - → Constraints are relaxed

[End of "Informed Search"]

24/06/2023

CSE314

⊕ 3 CPUs in xv6. ~~Keep~~ Make sure that a locking mechanism is present so that data is not corrupted for multiple CPUs.

⊕ How does OS start?

k/entry.S → k/start.c → k/main.c

CPU-0 ইম্পর্ট্যান্ট func-গুলো call হয়। তারপর started=1 করবে। In the meantime, CPU-1 & 2 will remain idle.

→ Initialization for memory

started=1 করা হলে CPU1 & 2 তাদের specific init করবে।

(#) Global variable হিসেবে কিছু lock আছে। এগুলো প্রতিটি initialization function-এর ভিতরে initialize করা হয়। [In k/main.c]

** void procinit() → একটি Process Table init করে। [k/proc.c]

```
void procinit()
{
  ----------
  initlock(....);
  ----------
}
```

** void userinit() → Setup first user process, running ONLY in CPU-0 [k/proc.c]

```
void userinit()
{
  // allocate a process
  // প্রত্যেক Process-এর
  // আলাদা Page table থাকে, সেগুলো init
  // xv6-এ "user" নামে আলাদা directory নেই,
  // তার root "/" থেকে start
  ----, initcode, ----
}
```

offline

proc.h
- proc struct
  - lock লাগাতে, কারণ কোন CPU-থেকে কোন Process run হচ্ছে

proc.c
- lock acquire, do sth, release lock [allocproc()]
  - Process state set করা/ state read করা
- New process create / child process create
  - allocproc()
  - fork()
- Process end হলে some cleanup → exit()

CPU - start থেকে যেখান কতগুলো tick → ticks
- 1 tick = 0.1 s
- tickslock acquire, read ticks, release lock

Include check, "extern" a declaration
- correct files, correct sequence

Multiple CPU-তে একই জায়গায় আসতে পারে কিনা, যদি পারে—তাহলে lock use করতে হবে।

args ::= type(s) and sequence ঠিক থাকতে হবে।