CS170 Operating Systems

Discussion Section

10/02/09
Me
Contact me

• Alice Tang
  – xtang@cs.ucsb.edu

• Prof. & me
  – cs170-admin@lists.cs.ucsb.edu

• Class (including Prof. & me)
  – cs170-users@lists.cs.ucsb.edu

• My Office Hour
  – Tue. 3pm-5pm & Fri. 2pm-5pm
  – CSIL Lab
What you can expect from me

• help you learn
  – If you are struggling
  – if you are not sure
  – if you have any questions
  – plz come and ask

• I don’t have all the answers
• But let’s try finding them!
You

• Grad/undergrad?
• Newcomer to UCSB?

• Office hour time conflict?

• Previous C programming experience?
• Previous Linux programming experience?
What do we discuss?

• mostly, **projects** & how principles taught in class implemented in projects
• Any course-related questions

• What I expect from you:
  – Read project requirement **in advance**
  – Start coding
  – Tell me what you feel the most difficult part
Project grading policy

• As stated in course homepage
  – no deadline extensions or late turn ins
  – Discussions are encouraged, but any actual copying of code will result in a 0 on the assignment

• How I will grade
  – Run tests
    • Functionality
    • Error cases
  – Read your code if necessary
    • comment when you need to make a point
    • use meaningful variable names
12 days left...

- 1\textsuperscript{st} project: write your own shell

- Start now!
Fork(), execve() & wait()

```
int main(int argc, char* argv[]) {

    int status;
    char* ls_args[2];
    ls_args[0] = ".";
    ls_args[1] = 0;
    if(fork() > 0) {
        /* Parent */
        waitpid(-1,&status,0);
        exit(status);
    } else {
        /* Child */
        execve("/bin/ls", ls_args,0);
    }

    return status;
}
```

• Why do we need such mechanism?
Slightly different...

#define TRUE 1

while (TRUE) {
    type_prompt();
    read_command(command, parameters);
    if (fork() != 0) {
        /* Parent code. */
        waitpid(-1, &status, 0);
    } else {
        /* Child code. */
        execve(command, parameters, 0);
    }
}
Steps for shell

1. print a prompt
2. Get the command line
3. Parse the command
   \textit{Command-name [parameter] [parameter]}
4. Find the file
   User’s \texttt{.login} file
   \textit{Set path=(./bin:/usr/bin)}
5. Prepare the parameters
6. Execute the command
   \texttt{Fork(), execve() & wait()}
Here comes the trouble...

- Input redirect <
- Output redirect >
- Pipe |
- Background implement &
Any comment, question or answer?