CSCI 3431 Operating Systems

Iteration 1 Report

Tsubasa Hirooka

Lianzhu Shi

Yiying Zhang

Yucheng Liu

1. Introduction

Our group decided to build up a simple UNIX shell, which basically has three functions: history, pipeline and tab completion. Due to the time consideration, we will try to finish these three functions first then we may start implement the database function. The shell would be built up through several C programs. And we may finally combine all the codes into a big C program.

2. Functions

- **History**: assign to Tsubasa Hirooka

  The history function would display all the commands the user has entered so far. Through this function, users can keep track on the commands they have entered and check if there are any problems. History can significantly save users' time; it simplifies spelling corrections and the repetition of complicated commands. If one user has entered a complex command, and whenever the user want to use this command again, he/she just need to enter "history", and then all the previous entered commands would be displayed on the screen. Users do not need to do any searching or worry about any spelling mistakes. Up till now, our shell can hold maximum 10 commands.

- **Pipeline**: assign to other three members

  The pipeline function is used to connect multiple commands together with the symbol "|". Pipes can be the best solution to deal with the I/O redirection. Users can enter several commands at the same line and pipes can connect all the commands together. All the programs in pipeline run at the same time, but at syntax, it would be arranged one

after another. This function makes our shell more user-friendly and easier to operate; users do not need to enter a new line after each command, which effectively changes the interface into a neat one.

- **Tab completion**: assign to Yiying Zhang and Yucheng Liu

  Rather than using tab to indent the space of our text, Tab completion would give "tab" a new function. In our UNIX shell, after users entering one or few letters and pressing the tab button, the shell would automatically do a search to find if there exists any command start with the letter(s) entered by user and show the searching results to users. Tab completion gives users a better solution to deal with the huge amounts of the commands in the UNIX shell. As we all know, there are probably hundreds of commands in a UNIX shell in order to bring out all the functions, and if users want to recite all the commands that would be wasted lots of time, but since we have tab completion, users just need to remember the first letter and then the tab button can do the rest for them, they just need to pick up the one they need.

3. Progress

We have started writing the codes for the history and pipe functions since November 2. Up until knowing the function may not work very well, we still spend our time on debugging the codes. Since the tab completion is the more difficult one, now we just have some ideas for how to implement it. We plan to start this function no later than November 12. After finishing all the coding and debugging parts, we may spend several days on combine all the codes into a big C program to make our project looks neat.

During that time period, we would double check if all the functions work well. Our schedule in details is shown at the end of this report.

4. Development

In the next iteration, first we will finish the history and pipe functions and then start doing the tab completion part. The tab completion function may take us 1 to 2 week to build it up and make it work. After we finish these three functions, we plan to add another function, database, into our shell to make the shell more useful.

5. Schedule

October 29, 2014 Having a group meeting to discuss the ideas about the project and choose the functions we want to implement

November 3, 2014 After searching some information about the shell, all the group members start writing some codes to implement our shell.

November 3, 2014 ~ November 6, 2014

Tsubasa works on the history; other group members work on the rest two functions.

November 7, 2014 Iteration 1 report due

November 12, 2014 Start working on the tab completion.

November 14, 2014 Basically fix the problems in pipe and history

November 23, 2014 Finish the tab completion

November 28, 2014 Combine all the codes into a big C program

Till the end of term Prepare for the presentation

6. Github Link

https://github.com/LYC1929/3431_OS_Project