Name: Tan Han Nguyen

NetID: TXN200004

Week 09 Lab 1

0. Log in

Pre-authentication banner message from server:

| University of Texas at Dallas

| Department of Computer Science

|

| Use of UTD Information Systems is subject to

| the UTD Information Security and Acceptable Use Policy.

|

| Pursuant to Texas Administrative Code 202:

| (1) Unauthorized use is prohibited;

| (2) Usage may be subject to security testing and monitoring;

| (3) Misuse is subject to criminal prosecution; and

| (4) No expectation of privacy except as otherwise provided by applicable

| privacy laws.

|

| ATTENTION: utdnetid != utdnetid@utdallas.edu (UTD != Google!)

|

| \*\*\*\*\* This system will require a connection to the GlobalProtect VPN startin

> g

| on the following dates:

|

| cslinux1.utdallas.edu - June 15, 2020

| cslinux2.utdallas.edu - June 22, 2020

|

| \*\*\*\*\* GlobalProtect VPN Instructions: https://www.utdallas.edu/oit/howto/vpn

> /

|

End of banner message from server

Keyboard-interactive authentication prompts from server:

| Password:

End of keyboard-interactive prompts from server

┌──────────────────────────────────────────────────────────────────────┐

│ • MobaXterm Personal Edition v24.2 • │

│ (SSH client, X server and network tools) │

│ │

│ ⮞ SSH session to txn200004@cslinux2.utdallas.edu │

│ • Direct SSH : ✓ │

│ • SSH compression : ✓ │

│ • SSH-browser : ✓ │

│ • X11-forwarding : ✓ (remote display is forwarded through SSH) │

│ │

│ ⮞ For more info, ctrl+click on help or visit our website. │

└──────────────────────────────────────────────────────────────────────┘

Last login: Wed Oct 16 21:26:41 2024 from 10.50.241.34

\*\*\*---\*\*\*---\*\*\*---\*\*\*---\*\*\*---\*\*\*

csgrads1.utdallas.edu - CentOS Linux 7.9

--All CS Graduate Students should use csgrads1--

cs1.utdallas.edu - CentOS Linux 7.9

cs2.utdallas.edu - CentOS Linux 7.9

\*\*\*---\*\*\*---\*\*\*---\*\*\*---\*\*\*---\*\*\*

This system is for use by CS students who need a general purpose Linux system

to complete homework assignments. Computationally or resource intensive

simulations will be throttled automatically.

Thank you,

CS Lab Manager

cs-labs@utdallas.edu

/scratch disk space can be used for temporary files.

All files within /scratch will be erased on a regular basis (Sunday 0300).

{cslinux2:~} mkdir week09Lab1; cd week09Lab1

1-2. Write myShell2.c and show the pipe segment

{cslinux2:~/week09Lab1} vim myShell2.c

{cslinux2:~/week09Lab1} cat myShell2.c

This part of the program has been cut for a more concise document. The original log is in the log file.

// pipe case

else if (strchr(line\_backup, '|') != NULL){

//Variable to use 2 pipes for multiple commands

int i = 0;

int pfd[2]; // Pipe file descriptors

int prev\_pfd[2]; // For holding the previous pipe's file descriptors

pid\_t pid;

char\* command; // Pointer to each individual command

char\* command\_array[MAX\_ARG]; // Array to store the command pointers

int command\_count = 0; // counter to keep track of command\_array

// Use strtok() to split line\_backup by ';'. tokenize the first command

command = strtok(line\_backup, "|");

while (command != NULL) {

// Trim leading and trailing spaces from the command

while (\*command == ' ') {

command++; // Remove leading spaces by move command pointer forward

}

char \*end = command + strlen(command) - 1;

while (end > command && \*end == ' ') {

\*end-- = '\0'; // Remove trailing space then move end pointer backward

}

// Duplicate and store the command in the array and increment the counter

command\_array[command\_count++] = strdup(command);

// Get the next command in the sequence using NULL to continue tokenizing the same string

command = strtok(NULL, "|");

}

// Null-terminate the array of commands

command\_array[command\_count] = NULL;

// Loop through all commands

while (command\_array[i] != NULL) {

// Create a pipe for all but the last command. Since the last one -> stdout

if (command\_array[i + 1] != NULL) {

if (pipe(pfd) == -1) {

perror("pipe");

exit(1);

}

}

// Fork the child process for the current command

pid = fork();

if (pid == -1) {

perror("fork");

exit(1);

}

if (pid == 0) { // Child process

//If not the first command, redirect stdin to the read end of the previous pipe

if (i > 0) {

close(prev\_pfd[1]); //close the write end of prev pipe

dup2(prev\_pfd[0], STDIN\_FILENO); // redirect stdin to read end of prev pipe

close(prev\_pfd[0]); // close read end of prev pip

}

//If not the last command, redirect stdout to the write end of the current pipe

if (command\_array[i + 1] != NULL) {

close(pfd[0]); //close the read end of the curr pipe

dup2(pfd[1], STDOUT\_FILENO); //redirect stdout to write end of curr pipe

close(pfd[1]); //close write end of curr pipe

}

//parse, expand, execute each command string

parse(command\_array[i], argv);

expand\_wildcards(argv, expanded\_argv);

//NOTE: NOT using execute() because it interferes with the child process here.

if(execvp(\*argv, argv) < 0){

printf("\*\*ERROR: exec failed\n");

exit(1);

}

} else { //Parent process

//Close the previous pipe in the parent

if (i > 0) {

close(prev\_pfd[0]);

close(prev\_pfd[1]);

}

//Move current pipe to previous then start new curr pipe in the next iteration

if (command\_array[i + 1] != NULL) {

prev\_pfd[0] = pfd[0];

prev\_pfd[1] = pfd[1];

}

// Wait for the child process to finish

waitpid(pid, NULL, 0);

}

i++; // Move to the next command

}

// Free memory allocated by strdup

for (int j = 0; j < command\_count; j++) {

free(command\_array[j]); // Free each command duplicated by strdup

}

printf("%s", prompt); // Prompt again

continue;

}

//OTHER CASES//

//expand wildcards

expand\_wildcards(argv, expanded\_argv);

execute(expanded\_argv); //child execute other commands

printf("%s", prompt); //prompt again

}

return 0;

}

3. Run myShell2 executable with some test cases

{cslinux2:~/week09Lab1} gcc myShell2.c -o myShell2 -std=gnu99

{cslinux2:~/week09Lab1} ./myShell2

{myShell2}$ date

Thu Oct 17 00:22:50 CDT 2024

{myShell2}$ whoami

txn200004

{myShell2}$ ps

PID TTY TIME CMD

25478 pts/2 00:00:00 bash

25714 pts/2 00:00:00 myShell2

25717 pts/2 00:00:00 ps

{myShell2}$ ps -aux | grep txn200004

root 25469 0.0 0.0 186940 5960 ? Ss 00:13 0:00 sshd: txn200004 [priv]

txn2000+ 25475 0.0 0.0 187252 2616 ? S 00:14 0:00 sshd: txn200004@pts/2

root 25476 0.0 0.0 186944 5964 ? Ss 00:14 0:00 sshd: txn200004 [priv]

txn2000+ 25567 0.0 0.0 186944 2416 ? S 00:14 0:00 sshd: txn200004@notty

{myShell2}$ ls -l

total 48

-rwx--x--x 1 txn200004 se 13808 Oct 17 00:22 myShell2

-rw------- 1 txn200004 se 14871 Oct 17 00:19 myShell2.c

{myShell2}$ ls -l | wc

3 20 119

{myShell2}$ ls -l /etc | head -n 10

total 2992

drwxr-xr-x 2 root root 4096 Jun 9 2014 a2ps

-rw-r--r-- 1 root root 15190 Jun 9 2014 a2ps.cfg

-rw-r--r-- 1 root root 2562 Jun 9 2014 a2ps-site.cfg

drwxr-xr-x. 3 root root 4096 Jan 5 2021 abrt

drwxr-xr-x 4 root root 4096 Apr 10 2018 acpi

-rw-r--r--. 1 root root 16 Apr 7 2016 adjtime

drwxr-xr-x. 2 root root 4096 May 10 2018 akonadi

-rw-r--r-- 1 root root 1554 Mar 10 2017 aliases

-rw-r--r--. 1 root root 12288 May 20 2020 aliases.db

{myShell2}$ ls -l /etc | head -n 10 | wc

10 83 552

{myShell2}$ ls -l /etc | tail -n 10

drwxr-xr-x. 6 root root 4096 Jan 5 2021 yum

-rw-r--r-- 1 root root 970 Oct 1 2020 yum.conf

-rw-r--r-- 1 root root 188 Apr 21 2015 yumex.conf

-rw-r--r-- 1 root root 32 Apr 21 2015 yumex.profiles.conf

drwxr-xr-x. 2 root root 4096 Aug 15 16:22 yum.repos.d

-rw-r--r-- 1 root root 252 Apr 7 2020 zlogin

-rw-r--r-- 1 root root 86 Apr 7 2020 zlogout

-rw-r--r-- 1 root root 375 Apr 7 2020 zprofile

-rw-r--r-- 1 root root 510 Apr 7 2020 zshenv

-rw-r--r-- 1 root root 1135 Apr 7 2020 zshrc

{myShell2}$ ls -l /etc | tail -n 10 | wc

10 90 613

{myShell2}$ ls -l /etc | tail -n 100 | tail -n 20 | head -n 10

drwxr-xr-x 2 root root 4096 May 16 2017 vpnc

-rw-r--r-- 1 root root 4479 May 15 2019 wgetrc

drwxr-xr-x. 2 root root 4096 Aug 13 2021 wpa\_supplicant

drwxr-xr-x. 7 root root 4096 Jan 9 2019 X11

drwxr-xr-x. 10 root root 4096 Mar 28 2024 xdg

-rw------- 1 root root 1001 Apr 1 2020 xinetd.conf

drwxr-xr-x. 2 root root 4096 May 20 2020 xinetd.d

drwxr-xr-x. 3 root root 4096 Jan 5 2018 xml

drwxr-xr-x 2 root root 4096 May 16 2017 xrdb

-rw-r--r-- 1 root root 619 Dec 28 2023 yp.conf

{myShell2}$ exit

End Lab and Exit

{cslinux2:~/week09Lab1} ls -l

total 48

-rwx--x--x 1 txn200004 se 13808 Oct 17 00:22 myShell2

-rw------- 1 txn200004 se 14871 Oct 17 00:19 myShell2.c

{cslinux2:~/week09Lab1} uname -a

Linux cslinux2.utdallas.edu 3.10.0-1160.119.1.el7.x86\_64 #1 SMP Tue Jun 4 14:43:51 UTC 2024 x86\_64 x86\_64 x86\_64 GNU/Linux

{cslinux2:~/week09Lab1} date

Thu Oct 17 00:25:03 CDT 2024

{cslinux2:~/week09Lab1} exit

logout

─────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────────

Session stopped

- Press <Return> to exit tab

- Press R to restart session

- Press S to save terminal output to file