**CS 35L Labs and Homeworks**

ls # lists your files

ls -l # lists your files in 'long format', which contains the exact size of the file, who owns the file and who has the right to look at it, and when it was last modified

ls -a # lists all files, including hidden files

ln -s <filename> <link> # creates symbolic link to file

touch <filename> # creates or updates your file

cat > <filename> # places standard input into file

more <filename> # shows the first part of a file (move with space and type q to quit)

head <filename> # outputs the first 10 lines of file

tail <filename> # outputs the last 10 lines of file (useful with -f option)

emacs <filename> # lets you create and edit a file

mv <filename1> <filename2> # moves a file

cp <filename1> <filename2> # copies a file

rm <filename> # removes a file

diff <filename1> <filename2> # compares files, and shows where they differ

wc <filename> # tells you how many lines, words and characters there are in a file

chmod -options <filename> # lets you change the read, write, and execute permissions on your files

gzip <filename> # compresses files

gunzip <filename> # uncompresses files compressed by gzip

gzcat <filename> # lets you look at gzipped file without actually having to gunzip it

lpr <filename> # print the file

lpq # check out the printer queue

lprm <jobnumber> # remove something from the printer queue

genscript # converts plain text files into postscript for printing and gives you some options for formatting

dvips <filename> # print .dvi files (i.e. files produced by LaTeX)

grep <pattern> <filenames> # looks for the string in the files

grep -r <pattern> <dir> # search recursively for pattern in directory

**# 1.2. Directory Commands.**

mkdir <dirname> # makes a new directory

cd # changes to home

cd <dirname> # changes directory

pwd # tells you where you currently are

**# 1.3. SSH, System Info & Network Commands.**

ssh user@host # connects to host as user

ssh -p <port> user@host # connects to host on specified port as user

ssh-copy-id user@host # adds your ssh key to host for user to enable a keyed or passwordless login

whoami # returns your username

passwd # lets you change your password

quota -v # shows what your disk quota is

date # shows the current date and time

cal # shows the month's calendar

uptime # shows current uptime

w # displays whois online

finger <user> # displays information about user

uname -a # shows kernel information

man <command> # shows the manual for specified command

df # shows disk usage

du <filename> # shows the disk usage of the files and directories in filename (du -s give only a total)

last <yourUsername> # lists your last logins

ps -u yourusername # lists your processes

kill <PID> # kills (ends) the processes with the ID you gave

killall <processname> # kill all processes with the name

top # displays your currently active processes

bg # lists stopped or background jobs ; resume a stopped job in the background

fg # brings the most recent job in the foreground

fg <job> # brings job to the foreground

ping <host> # pings host and outputs results

whois <domain> # gets whois information for domain

dig <domain> # gets DNS information for domain

dig -x <host> # reverses lookup host

wget <file> # downloads file

**# 2. Basic Shell Programming.**

**# 2.1. Variables.**

varname=value # defines a variable

varname=value command # defines a variable to be in the environment of a particular subprocess

echo $varname # checks a variable's value

echo $$ # prints process ID of the current shell

echo $! # prints process ID of the most recently invoked background job

echo $? # displays the exit status of the last command

export VARNAME=value # defines an environment variable (will be available in subprocesses)

array[0] = val # several ways to define an array

array[1] = val

array[2] = val

array=([2]=val [0]=val [1]=val)

array(val val val)

${array[i]} # displays array's value for this index. If no index is supplied, array element 0 is assumed

${#array[i]} # to find out the length of any element in the array

${#array[@]} # to find out how many values there are in the array

declare -a # the variables are treaded as arrays

declare -f # uses funtion names only

declare -F # displays function names without definitions

declare -i # the variables are treaded as integers

declare -r # makes the variables read-only

declare -x # marks the variables for export via the environment

${varname:-word} # if varname exists and isn't null, return its value; otherwise return word

${varname:=word} # if varname exists and isn't null, return its value; otherwise set it word and then return its value

${varname:?message} # if varname exists and isn't null, return its value; otherwise print varname, followed by message and abort the current command or script

${varname:+word} # if varname exists and isn't null, return word; otherwise return null

${varname:offset:length} # performs substring expansion. It returns the substring of $varname starting at offset and up to length characters

${variable#pattern} # if the pattern matches the beginning of the variable's value, delete the shortest part that matches and return the rest

${variable##pattern} # if the pattern matches the beginning of the variable's value, delete the longest part that matches and return the rest

${variable%pattern} # if the pattern matches the end of the variable's value, delete the shortest part that matches and return the rest

${variable%%pattern} # if the pattern matches the end of the variable's value, delete the longest part that matches and return the rest

${variable/pattern/string} # the longest match to pattern in variable is replaced by string. Only the first match is replaced

${variable//pattern/string} # the longest match to pattern in variable is replaced by string. All matches are replaced

${#varname} # returns the length of the value of the variable as a character string

\*(patternlist) # matches zero or more occurences of the given patterns

+(patternlist) # matches one or more occurences of the given patterns

?(patternlist) # matches zero or one occurence of the given patterns

@(patternlist) # matches exactly one of the given patterns

!(patternlist) # matches anything except one of the given patterns

$(UNIX command) # command substitution: runs the command and returns standard output

**# 2.2. Functions.**

# The function refers to passed arguments by position (as if they were positional parameters), that is, $1, $2, and so forth.

# $@ is equal to "$1" "$2"... "$N", where N is the number of positional parameters. $# holds the number of positional parameters.

functname() {

shell commands

}

unset -f functname # deletes a function definition

declare -f # displays all defined functions in your login session

**# 2.3. Flow Control.**

statement1 && statement2 # and operator

statement1 || statement2 # or operator

-a # and operator inside a test conditional expression

-o # or operator inside a test conditional expression

str1=str2 # str1 matches str2

str1!=str2 # str1 does not match str2

str1<str2 # str1 is less than str2

str1>str2 # str1 is greater than str2

-n str1 # str1 is not null (has length greater than 0)

-z str1 # str1 is null (has length 0)

-a file # file exists

-d file # file exists and is a directory

-e file # file exists; same -a

-f file # file exists and is a regular file (i.e., not a directory or other special type of file)

-r file # you have read permission

-r file # file exists and is not empty

-w file # your have write permission

-x file # you have execute permission on file, or directory search permission if it is a directory

-N file # file was modified since it was last read

-O file # you own file

-G file # file's group ID matches yours (or one of yours, if you are in multiple groups)

file1 -nt file2 # file1 is newer than file2

file1 -ot file2 # file1 is older than file2

-lt # less than

-le # less than or equal

-eq # equal

-ge # greater than or equal

-gt # greater than

-ne # not equal

if condition

then

statements

[elif condition

then statements...]

[else

statements]

fi

for x := 1 to 10 do

begin

statements

end

for name [in list]

do

statements that can use $name

done

for (( initialisation ; ending condition ; update ))

do

statements...

done

case expression in

pattern1 )

statements ;;

pattern2 )

statements ;;

...

esac

select name [in list]

do

statements that can use $name

done

while condition; do

statements

done

until condition; do

statements

done

**Lab #1**

1. man -k

commands used:

man man (and then scrolled through man page)

2. /usr/local/cs/bin/mv

/usr/local/cs/bin/cp

commands used:

which mv

which cp

3. ar: allows for creation and manipulation of archives, as well as extracting from them.

tr: used to translating or deleting letters, such as from lowercase to uppercase.

pr: formats files by adding pages and columns to make it look better when printed

commands used:

find /usr/bin -name "?"r

man ar

man tr

man pr

4. /usr/bin/emacs-24.3

commands used:

readlink -f /usr/bin/emacs

man readlink

5. /usr/bin/gcc: 4.8.5

gcc: 9.2.0

commands used:

gcc --version

/usr/bin/gcc --version

Different because gcc is actually in /usr/local/cs/bin, so it is distinct from /usr/bin/gcc.

6. chmod u+sx,o-w: u+sx part means to set the user ID for the user (temporarily giving it same permissions as superuser), which will allow execute permissions to be added. The o-w part means to remove write permissions from other users.

commands used:

man chmod

7. First 5 entries:

/usr/local/cs

/usr/local/cs/src/bison

/usr/local/cs/src/bison/RCS

/usr/local/cs/src/gnuplot

/usr/local/cs/src/gnuplot/RCS

There are a total of 175 directories with this property.

commands used:

man find

man wc

find /usr/local/cs -type d -mtime -28

find /usr/local/cs -type d -mtime -28 | wc

8. 303

commands used:

which find

cd /usr/bin

man grep

man wc

ls -la | grep "\->" | wc -l

Note: done on lnxsrv09

9. librom1394.so.0.3.0

commands used:

ls /usr/lib64 -lta

man ls

10. It gets its data from /usr/lib/locale/locale-archive

commands used:

man locale

man -k locale | grep -v -e "::" (grep to filter out results)

man localedef (then read through man localedef)

Lab #2:

Checked locale - not C locale, so used export LC\_ALL='C' to change it.

Verified that it changed by using locale again afterwards.

Then used sort /usr/share/dict/words > ~/words

To extract HTML page:

wget [CS35L assignment 2 link], where link was:

https://web.cs.ucla.edu/classes/fall19/cs35L/assign/assign2.html

tr commands: used cat assign2.html | [tr command] > output.txt

1) tr -c 'A-Za-z' '[\n\*]':

Deletes all lines that don't contain A-Z or a-z.

Replaces them with a newline.

2)tr -cs 'A-Za-z' '[\n\*]':

Adding -s option consolidates consecutive repeated newlines into one newline.

This makes the output more readable.

3) tr -cs 'A-Za-z' '[\n\*]' | sort:

This sorts the lines of what tr produces.

Capital letters come before lowercase letters and the sorting is alphabetical.

4) tr -cs 'A-Za-z' '[\n\*]' | sort -u:

Aadding the -u options eliminates any duplicate words.

5) tr -cs 'A-Za-z' '[\n\*]' | sort -u | comm - words:

Compare output of command #4 to words. Column 1 for words unique to tr output.

Column 2 is for words unique to words, column 3 is for words in both files.

6) tr -cs 'A-Za-z' '[\n\*]' | sort -u | comm -23 - words:

This command outputs only words unique to the output of the 4th tr command.

The -23 flag eliminates the 2nd and 3rd columns.

To make hwords:

1) Get HTML file

wget https://www.mauimapp.com/moolelo/hwnwdshw.htm # get HTML file

mv hwndshw.htm hwordsUnfinished # change file name

vim buildwords

2) Create script

grep '.\*</td>' | \ #isolate lines with td tags only

sed 's/<[^>]\*>//g' | \ #get rid of all HTML comments

tr "\`" "\'" | \ #replace ` with '

tr "A-Z" "a-z" | \ #all uppercase to lowercase

sed 's/^ \*//g' | \ #delete leading whitespace in lines

sed 's/?//g' | \ #delete all ? characters

grep -E "^[pk'mnwlhaeiou ]+$" | \ #delete all lines with non-hawaiian chars#!/bin/bash

sed 's/ /\n/g' | \ #replace all spaces with newlines

grep -E -v '^$' | \ #delete all blank lines

sort -u #sort and eliminate any duplicates

3) modify permissions and run script

chmod u+x buildwords

cat hwordsUnfinished | ./buildwords > hwords

ENGLISHCHECKER:

cat assign2.html | tr -cs 'A-Za-z' '[\n\*]' | tr '[:upper:]' '[:lower:]' | \

sort -u | comm -23 - words

Then pipe to wc -w to get # of words, or to badEng.txt to save as text file.

HAWAIIANCHECKER:

sed 's/<[^>]\*>//g' | tr "\`" "\'" | tr '[:upper:]' '[:lower:]' | \

tr -cs "A-Aa-z'" '[\n\*]' | sort -u | comm -23 - hwords

Then pipe to wc -w to get # of words, or to badHI.txt to save as text file.

Distinct misspelled words of assignment page using ENGLISHCHECKER: 37 words.

(Note: converted all letters in assignment page to lowercase).

Distinct misspelled words using HAWAIIANCHECKER: 515 words

Number of distinct words that ENGLISHCHECKER reports as misspelled but

HAWAIIANCHECKER does not: 2 words

Examples: lau, wiki

comm -12 used instead of -23 to get the number of matches.

Number of distinct words that HAWAIIANCHECKER reports as misspelled but

ENGLISHCHECKER does not: 477 words

Examples: your, yourself

comm -12 used instead of -23 to get the number of matches.

Homework:

#!/bin/bash

#poornames script

recursive\_mode=0 #false by default

directory=. #default directory

dir\_in\_second=0 #false by default

if [ "$1" ]; then

case "$1" in

-r)

recursive\_mode=1

dir\_in\_second=1

echo ' Recursive flag set'

;;

-?\*)

echo ' ' "$1" ': Invalid option, -r only, EXITING' >&2

exit 1

;;

\*)

echo ' Possible directory name: ' "$1"

directory="$1"

;;

esac

fi

if [ "$2" ] && [ "$dir\_in\_second" == 1 ]

then

directory="$2"

echo ' Directory set to ' "$directory"

fi

if [ "$directory" ]; then

if [ ! -d "$directory" ] || [ -L "$directory" ]; then

echo 'Directory does not exist or is not a directory, EXITING' >&2

exit 1

fi

fi

### Code execution here ###

set -f

IFS='

'

if [ "$recursive\_mode" == 0 ]; then

echo 'Non-recursive mode'

reqs\_1\_to\_4=`find "$directory" -maxdepth 1 | \

grep -v -E '^(.\*)\/([a-zA-Z\_\n][a-zA-Z.\_-]{0,13})$|^\.\.$|^\.$'`

req5=`find "$directory" -maxdepth 1 | sort -f | uniq -iD`

for $dir in reqs\_1\_to\_4

do

echo "$dir"

done

echo 'END'

total=`echo "$reqs\_1\_to\_4" && echo "$req5"`

total\_sorted=`echo "$total" | sort -u`

for dir in $total\_sorted

do

#if [ "$dir" != "$directory" ]; then

if [ -d "$dir" ]; then

echo "$dir"/

else

echo "$dir"

fi

#fi

done

echo 'DONE'

fi

if [ "$recursive\_mode" == 1 ]; then

echo 'Recursive mode'

reqs\_1\_to\_4=`find "$directory" | \

grep -v -E '^(.\*)\/([a-zA-Z\_][a-zA-Z.\_-]{0,13})$|^\.\.$|^\.$'`

req5\_all=

for dir1 in `find "$directory"`

do

if [ -d "$dir1" ]; then

req5=`find "$dir1" -maxdepth 1 | sort -f | uniq -iD`

if [ "$req5" ]; then

req5\_all=`echo "$req5\_all" && echo "$req5"`

fi

fi

done

total=`echo "$reqs\_1\_to\_4" && echo "$req5\_all"`

total\_sorted=`echo "$total" | sort -u`

for dir in $total\_sorted

do

#if [ "$dir" != "$directory" ]; then

if [ -d "$dir" ]; then

echo "$dir"/

else

echo "$dir"

fi

#fi

done

fi

IFS=''

**Lab #3:**

**Lab 3 Log**

Create shell script - copied the la script into a file and

made it an executable.

Verification that problem exists: running /usr/bin/ls -a -A gives the same

result as /usr/bin/ls -A -a. The . and .. are displayed.

ls Coreutils version: 8.22

To download the files:

Tarball: wget ftp://ftp.gnu.org/gnu/coreutils/coreutils-8.29.tar.xz

Tarball sig.: wget ftp://ftp.gnu.org/gnu/coreutils/coreutils-8.29.tar.xz.sig

GNU keyring: wget https://ftp.gnu.org/gnu/gnu-keyring.gpg

Verfiy signature:

gpg --verify --keyring ./gnu-keyring.gpg coreutils-8.29.tar.xz.sig

It outputs 3 good signature statements, and then the following statement:

WARNING: This key is not certified with a trusted signature!

gpg: There is no indication that the signature belongs to the owner.

This occurs because the key has not been verified, as it could have been

generated by anyone.

Unzipped file:

tar -xJvf coreutils-8.29.tar.xz

Made directory for installtion:

~/cuinstall (core util install)

Installing it:

./configure --prefix=[absolute path].../cuinstall

make

make install

Then to reproduce the bug in my newly installed coreutils:

/u/ma/ugrad/danning/cslab/cuinstall/bin/ls -a -A

This command prints out ., .., bin, libexec, and share, so the bug is there.

Then download patch:

In ~/coreutils-8.29, I created a file called lsfix.patch.

Then I opened the patch file and copied the diff file portion into a file

called lsfix.patch.

patch -p1 < lsfix.patch

Errors:

Hunk #1 FAILED at 4.

1 out of 1 hunk FAILED -- saving rejects to file NEWS.rej

It says the file tests/local.mk and tests/ls/a-option.sh were patched.

Then when I type make, it returns the error: 'automake-1.15' is missing on

your system. THis occurs because local.mk is modified, which is related to

configure.ac. Thus, configure needs to be rerun, or else automake won't work.

To solve this issue, the command autoreconf -fi must be used. This remakes

the build system files: -f forces all files to be considered obsolete and

-i looks for any missing files and copies them.

Then use make again.

To test the new version of ls:

/u/ma/ugrad/danning/cslab/coreutils-8.29/src/ls -a -A

This command does not print out . and .., confirming that the patch worked.

**Shuf.py**

#!/usr/bin/env python3

import random, sys, argparse, string

low\_val = None #int, lower value

high\_val = None #int, high value

print\_amt\_readin = -1

repeat\_lines = False #can lines be repeated

read\_from\_stdin = True #read from stdin or not

input\_range\_option = False #lines come from lo to hi, inclusive

to\_shuffle\_contents = "" #string separated by newlines

def extract\_number(alphanum):

number\_string=""

for char in alphanum:

if(char.isdigit()):

number\_string = number\_string + char

else:

break

return int(number\_string)

def create\_shuffled(to\_shuffle\_contents):

to\_shuffle\_list = to\_shuffle\_contents.split("\n") #convert to list

input\_range\_amount = len(to\_shuffle\_list)

print\_amt = print\_amt\_readin

if input\_range\_amount < print\_amt\_readin \

or (print\_amt\_readin == -1 and repeat\_lines == False):

#if less or not provided, only print the amount you have have

print\_amt = input\_range\_amount

if repeat\_lines == False:

shuffled\_list = random.sample(to\_shuffle\_list, print\_amt)

#random.sample() takes it without replacement

for item in shuffled\_list:

print(item)

elif repeat\_lines == True:

if print\_amt\_readin != -1:

for i in range(0, print\_amt\_readin):

print(random.choice(to\_shuffle\_list))

elif print\_amt\_readin == -1:

while True:

print(random.choice(to\_shuffle\_list))

parser = argparse.ArgumentParser()

parser.add\_argument("-i", "--input-range", action="store", dest="input\_range",

help="treat each number in INPUT\_RANGE as an input line")

parser.add\_argument("-n", "--head-count", action="store",

dest="output\_count", help="output at most OUTPUT\_COUNT lines")

#read in value until first nondigit char

parser.add\_argument("-r", "--repeat", action="store\_true", default=False,

help="output lines can be repeated")

parser.add\_argument("FILE", nargs='?',

help="With no FILE, or when FILE is -, read standard input.")

try:

args = parser.parse\_args()

except:

sys.exit(1)

file\_name = args.FILE

repeat\_lines = args.repeat

if args.input\_range: #-i or --input-range

try:

dash\_index = args.input\_range.index('-')

except:

print("shuf: invalid inptut range: " + args.input\_range)

sys.exit(1)

read\_from\_stdin = False

input\_range\_option = True

# first occurence of -

try:

low\_val = int(args.input\_range[0:dash\_index])

high\_val = int(args.input\_range[dash\_index+1:])

if(low\_val > high\_val):

print("shuf: invalid input range: " + args.input\_range)

sys.exit(1)

for i in range(low\_val, high\_val+1):

to\_shuffle\_contents += (str(i)+"\n")

to\_shuffle\_contents = to\_shuffle\_contents[:-1]

except:

print("shuf: invalid input range")

sys.exit(1)

if args.output\_count: #-n or --head-count

try:

print\_amt\_readin = extract\_number(args.output\_count)

except:

print("shuf: invalid line count: " + args.output\_count)

sys.exit(1)

if args.FILE:

if input\_range\_option:

print("shuf: extra operand " + args.FILE)

print("Try 'shuf --help' for more information.")

sys.exit(1)

if args.FILE == "-":

read\_from\_stdin = True

else:

try:

text\_file = open(args.FILE, "r")

to\_shuffle\_contents = text\_file.read()

to\_shuffle\_contents = to\_shuffle\_contents[:-1]

read\_from\_stdin = False #no longer need to read

except:

print("shuf: " + args.FILE + ": No such file or directory")

sys.exit(1)

if read\_from\_stdin:

to\_shuffle\_contents = sys.stdin.read()

# += not needed because read() gobbles up everything

to\_shuffle\_contents = to\_shuffle\_contents[:-1] #remove \n at end

create\_shuffled(to\_shuffle\_contents)

**HW #3:**

1. The performance of ls improved: Prepend the "time" command to ls, and one

gets the following run times for running ls on /usr/bin:

time ./cuinstall/bin/ls /usr/bin outputs a real time of 0.060 s

time ./coreutils-8.29/src/ls /usr/bin outputs a real time of 0.056 s

Examining the patch file, we see that the patch removes a case 'A': statement,

which speeds up the program. Also, in general a bug has been fixed, which

also increases the performance of the program (in a correctness sense).

2. If in a program or script, you use the -a -A bug of the previous version of

ls on purpose to override the -A command so that . and .. are always listed,

there may be issues that arise from the upgrade. Or, if you have a program

that uses -A and -a in combination interchangeably, the program's behavior

will change because -a and -A are no longer be interchangeable.

3. It throws an IndexError: list index out of range. This occurs because

len(seq) = 0, so it tries to return seq[0], which is out of bounds because

seq is empty.

4. It throws a SyntaxError: invalid syntax, on line 65 of the file. This is

because in Python 3, error handling syntax is except IOError as err:

5. I ran my script with multiple test cases using Python 2 instead of 3 and

there did not appear to be any visible differences.

**Lab #4:**

Got the tar file using wget and unzip it:

wget http://web.cs.ucla.edu/classes/fall19/cs35L/assign/coreutils-with-bug.tar.gz

tar -xzvf coreutils-with-bug.tar.gz

Then navigate to unzipped folder and install coreutils:

mkdir cubug

cd coreutils-with-bug

./configure --prefix=[aboslute path...]/cubug

make

make install

When make is run, it displays an error: that there are conflicting types for

futimens. The same error pops up when make install is run.

To download and apply patch to fix this issue:

wget http://web.cs.ucla.edu/classes/fall19/cs35L/assign/coreutils.diff

patch -p0 < coreutils.diff

-p0 option was used so that the file path is not modified.

Then remake the installation:

make

There is no longer an error about futimens because the patch file renames

futimens to coreutils\_futimens. (Found this by examining the diff file).

It also does the same for a function called tee: renames to coreutils\_tee.

As a result, there's no longer a name conflict.

Because the installation failed earlier, it didn't actually install

coreutils into cubug. Thus, we need to redo the installation steps.

./configure --prefix=[aboslute path...]/cubug

make

make install

Reproducing the error:

tmp=$(mktemp -d)

cd $tmp

touch -d '1918-11-11 11:00 GMT' wwi-armistice-cs35L

touch now

sleep 1

touch now1

TZ=UTC0 ~/cslab/cubug/bin/ls -lt --full-time wwi-armistice-cs35L now now1

Output:

-rw-r--r-- 1 danning maugrad 0 1918-11-11 11:00:00.000000000 +0000 wwi-armistice-cs35L

-rw-r--r-- 1 danning maugrad 0 2019-10-22 04:32:16.745623119 +0000 now1

-rw-r--r-- 1 danning maugrad 0 2019-10-22 04:32:08.200360731 +0000 now

wwi-armistice-cs35L should be the last line of output, but it is erroneously

the first.

To figure out what's causing this error, I used gdb:

gdb ~/cubug/bin/ls

To find a function to break on:

info functions

For ls.c, I found the functions compare\_ctime and compare\_mtime, which I

think could cause the problem. Thus, I set breakpoints at both locations.

break compare\_ctime

break compare\_mtime

Then, I ran the command:

run -lt --full-time

It stopped on compare\_mtime, and by stepping through compare\_mtime, I found

that it calls another function, timespec\_cmp, so I added a breakpoint for

that too. Then I ran the program over from the start.

By stepping through code one line at a time, I found the following code in

timespec\_cmp:

int diff = a.tv\_sec - b.tv\_sec;

return diff ? diff : a.tv\_nsec - b.tv\_nsec;

I would guess overflow is occurring in the first line when subtraction occurs.

To test this, I used info local to monitor the value of diff as I stepped

through the program. At first, it was a positive value, but the last value

of diff before the comparisons finshed was -7, which makes no sense.

A potential fix to this program is just to compare the times directly and

return an appropriate value based on the comparison. GDB tells me that

these lines of code are in .../lib/timespec.h, lines 48-49.

Editing timespec.h: make a copy first

cd ~/coreutils-with-bug

cp lib/timespec.h timespec.h

emacs timespec.h

The function timespec\_cmp is given below:

timespec\_cmp (struct timespec a, struct timespec b)

{

int diff = a.tv\_sec - b.tv\_sec;

return diff ? diff : a.tv\_nsec - b.tv\_nsec;

}

Following the comment above the function, I modified it to the following:

timespec\_cmp (struct timespec a, struct timespec b)

{

if(a.tv\_sec > b.tv\_sec)

return 1;

else if(a.tv\_sec < b.tv\_sec)

return -1;

return a.tv\_nsec - b.tv\_nsec;

}

Then, create the diff file:

diff -u lib/timespec.h timespec.h > lab4.diff

Open file in emacs, and change the 2nd line from timespec.h to lib/timespec.h

so that when the diff file is applied, it'll modify the right file.

Then, used C-x 4 a to access the changelog, and added the following note:

2019-10-26 Danning Liu Yu <danning@lnxsrv07.seas.ucla.edu>

\* timespec.h (timespec\_cmp): Fixed error where overflow can

potentially occur if the difference between the times is very large.

Replaced with direct comparison. Either -1 or 1 is returned

depending on which timestamp is bigger. If they are equal with

respect to their seconds value, return difference in nanoseconds.

Then apply the patch, using same procedure as before:

patch -p0 < lab4.diff

make

Then redo the same steps that displayed the bug, except this time use the

patched ls command in coreutils-with-bug. This produces the following output:

-rw-r--r-- 1 danning maugrad 0 2019-10-22 17:32:12.868193895 +0000 now1

-rw-r--r-- 1 danning maugrad 0 2019-10-22 17:32:06.902011230 +0000 now

-rw-r--r-- 1 danning maugrad 0 1918-11-11 11:00:00.000000000 +0000 wwi-armistice-cs35L

The extremely old file now correctly appears at the bottom.

Creating wwi-armistice-cs35L, now, and now1 in my home directory and issuing

ls gives the following result:

-rw-r--r-- 1 danning maugrad 0 2054-12-17 17:28:16.000000000 +0000 wwi-armistice-cs35L

-rw-r--r-- 1 danning maugrad 0 2019-10-22 17:34:42.812352000 +0000 now1

-rw-r--r-- 1 danning maugrad 0 2019-10-22 17:34:36.697785000 +0000 now

This indicates that the SEASnet ls does not work correctly either. In fact,

it exhibits a different issue, where the year 1918 incorrectly shows up as

2054. This is because when the file is created, the date is written as a

32-bit unsigned integer, but when it is read, ls interprets it as a 32-bit

signed integer, thus resulting in overflow.

**Makefile:**

CC=gcc

sfrob: sfrob.c

$(CC) -o sfrob -fsanitize=address -fsanitize=undefined \

-static-libubsan -static-libasan -g -W -Wall -pedantic sfrob.c

clean:

rm -f sfrob

valgrind:

$(CC) -o sfrob -g -W -Wall -pedantic sfrob.c

sfrob.c:

#include <stdio.h>

#include <stdlib.h>

/\* Function Declarations \*/

int frobcmp(char const \*a, char const \*b);

int compare(const void\* a, const void\* b);

void printAllStrings(char \*\*arrOfStrings, int numberOfStrings);

/\* Function Implementations \*/

int frobcmp(char const \*a, char const \*b){

int alength;

int blength;

int alengthPreCompute;

int blengthPreCompute;

/\* get the lengths of a and b \*/

for(alengthPreCompute=0; a[alengthPreCompute] != ' '; alengthPreCompute++){}

for(blengthPreCompute=0; b[blengthPreCompute] != ' '; blengthPreCompute++){}

for(alength=0, blength = 0;

a[alength] != ' ' && b[blength] != ' ';

alength++, blength++){

if((a[alength] ^ 42) < (b[blength] ^ 42)){

return -1;

}

if((a[alength] ^ 42) > (b[blength] ^ 42)){

return 1;

}

}

/\* if exited for-loop, then equal up to certain point \*/

if(alengthPreCompute < blengthPreCompute){

return -1;

}

else if(alengthPreCompute > blengthPreCompute){

return 1;

}

/\* at this point, alength == blength \*/

return 0;

}

int compare(const void\* a, const void\* b){

return frobcmp(\*(char\*\*)(a), \*(char\*\*)(b));

}

void printAllStrings(char \*\*arrOfStrings, int numOfStrings){

int i;

/\* const int alwaysTrue = 1; \*/

for(i = 0; i<=numOfStrings; i++){

int j = 0;

while(arrOfStrings[i][j] != ' '){

if(putchar(arrOfStrings[i][j]) == EOF){

fprintf(stderr, "Error occurred when printing characters\n");

exit(1);

}

j++;

}

if(putchar(' ') == EOF){

fprintf(stderr, "Error occurred when printing characters\n");

exit(1);

}

}

}

/\* Main \*/

int main(){

int ch = 0;

int allStringsIndex = -1;

int addNewString = 1;

int alwaysTrue = 1;

/\* 1 for true (default), 0 for false \*/

int newStringIndex = 0;

char\* newString = NULL;

int i;

char\*\* allStrings = (char\*\*)(malloc(sizeof(char\*)));

if(allStrings == NULL){

fprintf(stderr, "Error in allocating memory to hold arr of strings\n");

exit(1);

}

/\* read in strings \*/

while(alwaysTrue){

ch = getchar();

if(ferror(stdin)){

fprintf(stderr, "Error when reading file\n");

exit(1);

}

if(feof(stdin)){

break;

}

if(addNewString){

/\* start of new string \*/

allStringsIndex++;

/\* expand allStrings to hold another ptr \*/

allStrings = (char\*\*)(realloc(allStrings, (allStringsIndex+1)\*sizeof(char\*)));

if(allStrings == NULL){

fprintf(stderr, "Error in allocating memory to hold arr of strings\n");

exit(1);

}

/\* make new ptr for strings, allocate size for 1 char \*/

newString = (char\*)(malloc(sizeof(char)));

if(newString == NULL){

fprintf(stderr, "Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = ch;

allStrings[allStringsIndex] = newString;

addNewString = 0;

newStringIndex++;

if(ch == ' '){

/\* space acts as "null byte" \*/

addNewString = 1;

newStringIndex = 0;

newString = NULL;

continue;

}

}

else if (!addNewString){

newString = (char\*)(realloc(newString, (newStringIndex+1)\*sizeof(char)));

if(newString == NULL){

fprintf(stderr, "Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = ch;

allStrings[allStringsIndex] = newString;

newStringIndex++;

if(ch == ' '){

/\* space acts as "null byte" \*/

addNewString = 1;

newStringIndex = 0;

newString = NULL;

}

}

}

if(allStringsIndex != -1 && newStringIndex != 0

&& allStrings[allStringsIndex][newStringIndex-1] != ' '){

/\* last char is not space \*/

newString = (char\*)(realloc(newString, (newStringIndex+1)\*sizeof(char)));

if(newString == NULL){

fprintf(stderr, "Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = ' ';

allStrings[allStringsIndex] = newString;

}

qsort(allStrings, allStringsIndex+1, sizeof(char\*), compare);

printAllStrings(allStrings, allStringsIndex);

/\* clean up allocated memory \*/

for(i = 0; i<=allStringsIndex; i++){

free(allStrings[i]);

}

free(allStrings);

exit(0);

}

**Lab #5:**

Lab Portion (lab.txt):

First, I wrote tr2b.c as specified, using getchar to read in bytes from

stdin and then ouputting them to stdout using putchar. These functions

come from stdio.h. I used an array of size 256 to map one character to

another for transliteration purposes, essentially a hash table. If

there were an incorrect number of arguments, duplicate elements in set

1, or if the length of set 1 was not equal to the length of set 2, I

reported the error to stderr and exited with exit code 1. Also, if

errors occurred during the reading or printing process, I reported the

error to stderr and exited with exit code 1. When printing, I checked

if the array contained a conversion from one character to another. If

so, print out the converted character; otherwise, print out the

original character.

Then, I wrote tr2u.c, with the only change being using read and write

from unistd.h instead of getchar and putchar. These functions are

system calls. Instead of buffered reading, I forced the program to read

and write bytes 1 at a time. The possible error conditions were the

same as for tr2b.c.

I then got a file with a size of 5,000,000 bytes using:

head --bytes=5000000 /dev/urandom > bigfile.txt

Then, I issued the following command to invoke tr2b for copying from one

file to another.

strace -c ./tr2b 'ABC' 'abc' < bigfile.txt > output1.txt

The strace command reported a total of 1,250 system calls for tr2b.

Then I ran the same command, but with tr2b changed to tr2u:

strace -c ./tr2u 'ABC' 'abc' < bigfile.txt > output2.txt

The number of system calls for tr2u is 10,000,026, which is dramatically

higher than that of tr2b (1,250 calls). As expected, a 5 million byte file

results in roughly 5 million calls each for write and read, as I am reading

and writing 1 byte at a time.

Then, I tested writing to terminal:

strace -c ./tr2b 'ABC' 'abc' < bigfile.txt

strace -c ./tr2u 'ABC' 'abc' < bigfile.txt

This gave a total of 20,495 system calls for tr2b and 1,000,026 calls for

tr2u. The number of system calls dramatically increased for tr2b for writing

to to the terminal possibly because the terminal buffer can hold less before

it must display all its contents. The number of system calls for tr2u stayed

the same.

Timing each of the 4 commands I made earlier:

1) time ./tr2b 'ABC' 'abc' < bigfile.txt > output3.txt

2) time ./tr2u 'ABC' 'abc' < bigfile.txt > output4.txt

3) time ./tr2b 'ABC' 'abc' < bigfile.txt

4) time ./tr2u 'ABC' 'abc' < bigfile.txt

The commands give the following time outputs:

1)

real 0m0.271s

user 0m0.220s

sys 0m0.008s

2)

real 0m12.537s

user 0m3.225s

sys 0m9.269s

3)

real 0m5.915s

user 0m0.271s

sys 0m0.107s

4)

real 0m12.326s

user 0m3.300s

sys 0m8.298s

As expected, tr2b is much faster than tr2u when reading in and writing out

data. When the output is the terminal window instead, tr2b loses a bit of

its advantage, due to the increased number of system calls needed when

writing to the terminal. However, it is still faster. The time it takes to

run tr2u for writing to a file and to the terminal is very similar, which

makes sense, as the same number of system calls is made.

Overall, we can see a correlation between the number of system calls and how

long it takes a program to run.

Homework Portion (report.txt):

Running sfrob and sfrobu with various line counts:

0 lines 100 lines

sfrob real 0m0.003s real 0m0.002s

user 0m0.003s user 0m0.002s

sys 0m0.001s sys 0m0.000s

sfrobu real 0m0.011s real 0m0.013s

user 0m0.004s user 0m0.003s

sys 0m0.007s sys 0m0.010s

10,000 lines 1,000,000 lines

sfrob real 0m0.006s real 0m0.549s

user 0m0.004s user 0m0.533s

sys 0m0.001s sys 0m0.015s

sfrobu real 0m0.059s real 0m7.733s

user 0m0.034s user 0m3.786s

sys 0m0.025s sys 0m3.947s

For sfrob, the correspondence between # of lines (N) and runtime (t) is

roughly t = (9.15E-8)(NlogN), where logN here indicates base 10.This makes

sense, as sfrob uses quicksort to sort the lines, and quicksort has a typical

O(NlogN) runtime.

For sfrobu, the runtime is also dependent on NlogN: t = (1.28E-6)(NlogN),

which makes sense because sfrobu uses quicksort as well. However, due to its

use of sys calls, it has a higher constant factor (1.28E-6 versus 9.15E-8),

causing it to be slower.

Using the strace command, malloc, realloc, and free seem to call the following

library functions:

malloc: Either brk or mmap, depending on if a small block of new memory is

allocated (brk is used) or a large block of memory is allocated (mmap used).

If brk is used, its argument is an address location, which should be the new

end of the program data segment. If NULL is used as the address location, it

returns the current location of the program break, which can then be used

to calculate the new desired location.

If mmap is used, it takes in a hint to where the block of memory should be

mapped (can be NULL, in which case the kernel decides where to place the

block of memory), the desired size of the block of memory, the

read/write/execute/access permissions of this memory segment, flags to

determine if this memory mapping should be visible to other processes, the

file to map this block of memory to (-1 in our case), and any offset, if

needed (0 in our case). For sfrobu, the rwx flag is

PROT\_READ|PROT\_WRITE and the mapping flag is MAP\_PRIVATE|MAP\_ANONYMOUS.

realloc: mmap is used. Everything is the same as what was described for mmap

above, except for the size: it is the updated desired size of the block of

memory. All other arguments remain the same.

free: munmap is used. It takes in an address to the block of memory to free

and how bytes should be freed. It then returns 0 upon success.

**Tr2b.c:**

#include <stdlib.h>

#include <string.h>

#include <stdio.h>

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

if(argc != 3){

fprintf(stderr, "Error: incorrect number of arguments\n");

exit(1);

}

if(strlen(argv[1]) != strlen(argv[2])){

fprintf(stderr, "Error: sets 1 and 2 must be the same length\n");

exit(1);

}

char\* fromChars = argv[1];

char\* toChars = argv[2];

unsigned char convertTable[256] = {'\0'};

/\* to hold all 7 bit ASCII characters \*/

size\_t i;

for(i = 0; i<strlen(fromChars); i++){

if(convertTable[(unsigned char)fromChars[i]] == '\0'){

convertTable[(unsigned char)fromChars[i]] = toChars[i];

}

else if(convertTable[(unsigned char)fromChars[i]] != '\0'){

fprintf(stderr, "Duplicate element(s) in from set\n");

exit(1);

}

}

int c;

while((c = getchar()) != EOF){

;

if(feof(stdin)){

break;

}

if(ferror(stdin)){

fprintf(stderr, "Error occurred when reading from stdin\n");

}

if(convertTable[c] != '\0'){

putchar(convertTable[c]);

if(ferror(stdout)){

fprintf(stderr, "Error when printing characters\n");

exit(1);

}

}

else{

putchar(c);

if(ferror(stdout)){

fprintf(stderr, "Error when printing characters\n");

exit(1);

}

}

}

exit(0);

}

**Tr2u.c:**

#include <stdlib.h>

#include <string.h>

#include <unistd.h>

/\* Danning Yu \*/

/\* 305087992 \*/

/\* Using write and read \*/

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

if(argc != 3){

write(STDERR\_FILENO, "Error: incorrect number of arguments\n", 37);

exit(1);

}

if(strlen(argv[1]) != strlen(argv[2])){

write(STDERR\_FILENO, "Error: sets 1 and 2 must be the same length\n", 44);

exit(1);

}

char\* fromChars = argv[1];

char\* toChars = argv[2];

char convertTable[256] = {'\0'};

/\* to hold all 7 bit ASCII characters \*/

size\_t i;

for(i = 0; i<strlen(fromChars); i++){

if(convertTable[(unsigned char)fromChars[i]] == '\0'){

convertTable[(unsigned char)fromChars[i]] = toChars[i];

}

else if(convertTable[(unsigned char)fromChars[i]] != '\0'){

write(STDERR\_FILENO, "Duplicate element(s) in from set\n", 33);

exit(1);

}

}

unsigned char charBuffer[1];

unsigned char outputBuffer[1];

while(read(STDIN\_FILENO, charBuffer, 1) > 0){

// if(convertTable[(unsigned char)charBuffer[0]] != '\0'){

// outputBuffer[0] = convertTable[(unsigned char)charBuffer[0]];

if(convertTable[charBuffer[0]] != '\0'){

outputBuffer[0] = convertTable[charBuffer[0]];

if(write(STDOUT\_FILENO, outputBuffer, 1) < 0){

write(STDERR\_FILENO, "Error when writing characters\n", 30);

exit(1);

}

}

else{

if(write(STDOUT\_FILENO, charBuffer, 1) < 0){

write(STDERR\_FILENO, "Error when writing characters\n", 30);

exit(1);

}

}

}

exit(0);

}

**Sfrobu.c (Lab 5):**

#include <stdlib.h>

#include <stdio.h>

#include <sys/types.h>

#include <sys/stat.h>

#include <unistd.h>

#include <string.h>

#include <ctype.h>

/\* Function Declarations \*/

int frobcmp(const unsigned char \*a, const unsigned char \*b);

int compare(const void\* a, const void\* b);

void printAllStrings(unsigned char \*\*arrOfStrings, int numberOfStrings);

/\* For -f option \*/

int case\_insensitive = 0;

/\* Function Implementations \*/

int frobcmp(const unsigned char \*a, const unsigned char \*b){

int alength;

int blength;

int alengthPreCompute;

int blengthPreCompute;

/\* get the lengths of a and b \*/

for(alengthPreCompute=0; a[alengthPreCompute] != ' '; alengthPreCompute++){}

for(blengthPreCompute=0; b[blengthPreCompute] != ' '; blengthPreCompute++){}

for(alength=0, blength = 0; a[alength] != ' ' && b[blength] != ' ';

alength++, blength++){

if(case\_insensitive){

if( (toupper( a[alength] ^ 42 ) ) < (toupper( b[blength] ^ 42 ) ) ){

return -1;

}

if( (toupper(a[alength] ^ 42)) > (toupper(b[blength] ^ 42)) ){

return 1;

}

}

else{

if((a[alength] ^ 42) < (b[blength] ^ 42)){

return -1;

}

if((a[alength] ^ 42) > (b[blength] ^ 42)){

return 1;

}

}

}

/\* if exited for-loop, then equal up to certain point \*/

if(alengthPreCompute < blengthPreCompute){

return -1;

}

else if(alengthPreCompute > blengthPreCompute){

return 1;

}

/\* at this point, alength == blength \*/

return 0;

}

int compare(const void\* a, const void\* b){

return frobcmp(\*(unsigned char\*\*)(a), \*(unsigned char\*\*)(b));

}

void printAllStrings(unsigned char \*\*arrOfStrings, int numOfStrings){

int i;

unsigned char charBuffer[1];

unsigned char spaceCharBuffer[1] = {' '};

/\* const int alwaysTrue = 1; \*/

for(i = 0; i<numOfStrings; i++){

int j = 0;

while(arrOfStrings[i][j] != ' '){

charBuffer[0] = arrOfStrings[i][j];

// if(putchar(arrOfStrings[i][j]) == EOF){

if(write(STDOUT\_FILENO, charBuffer, 1) < 0){

fprintf(stderr, "Error occurred when printing characters\n");

exit(1);

}

j++;

}

if(write(STDOUT\_FILENO, spaceCharBuffer, 1) < 0){

fprintf(stderr, "Error occurred when printing characters\n");

exit(1);

}

}

}

/\* Main \*/

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

/\* Read in arguments \*/

if(argc > 2){

fprintf(stderr, "Incorrect number of arguments\n");

exit(1);

}

if(argc == 2){

if(strlen(argv[1]) != 2){

fprintf(stderr, "Bad argument, only -f allowed\n");

exit(1);

}

if(argv[1][0] == '-' && argv[1][1] == 'f'){

// printf("Case insensitive option\n");

case\_insensitive = 1;

//how about if(argc == 2 && argv[1] != "-f")?

}

else {

fprintf(stderr, "Bad argument, only -f allowed\n");

exit(1);

}

}

/\* Read in text from stdin \*/

struct stat inputFile;

int fileSize = 0;

unsigned char\*\* allStrings = NULL;

unsigned char\* inputFileContents = NULL;

int numWords = 0;

int addedSpaceAtEnd = 0;

int alwaysTrue = 1;

unsigned char inBuffer[1];

int wordIndex = 0;

int startOfWord = 1;

int lastWordSize = 0;

int addNewString = 1;

int newStringIndex = 0;

unsigned char\* newString = NULL;

int zeroSizeInitialFile = 0;

if(fstat(STDIN\_FILENO, &inputFile) < 0){

fprintf(stderr, "Error when reading in from stdin\n");

exit(1);

}

// if(S\_ISREG(inputFile.st\_mode)){

// printf("regular file\n");

// }

// else if(S\_ISFIFO(inputFile.st\_mode)){

// printf("Piped in\n");

// }

if(inputFile.st\_size == 0){

// printf("Nothing to sort.\n");

zeroSizeInitialFile = 1;

}

if(S\_ISREG(inputFile.st\_mode) && inputFile.st\_size != 0){

fileSize = inputFile.st\_size;

// printf("File size is %d\n", fileSize);

inputFileContents = (unsigned char\*)(

malloc(fileSize\*sizeof(unsigned char)));

if(inputFileContents == NULL){

fprintf(stderr,

"Error in allocating memory to hold input file\n");

exit(1);

}

if(read(STDIN\_FILENO, inputFileContents, fileSize) < 0){

fprintf(stderr, "Error when reading in from file\n");

}

for(int i = 0; i<fileSize; i++){

if(inputFileContents[i] == ' '){

numWords++;

// printf("Now %d words\n", numWords);

}

}

if(inputFileContents[fileSize-1] != ' '){

// printf("Adding trailing space\n");

numWords++; //missing trailing space

addedSpaceAtEnd = 1;

fileSize = fileSize + 1;

inputFileContents = (unsigned char\*)(

realloc(inputFileContents, fileSize\*sizeof(unsigned char)));

inputFileContents[fileSize-1] = ' ';

}

allStrings = (unsigned char\*\*)(

malloc(numWords\*sizeof(unsigned char\*)));

if(allStrings == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

// printf("File size is %d\n", fileSize);

for(int i = 0; i<fileSize; i++){

// printf("Analyzing :%c:\n", inputFileContents[i]);

if(startOfWord){

allStrings[wordIndex] = &inputFileContents[i];

wordIndex++;

startOfWord = 0;

lastWordSize = 1;

// printf("Start of word at %d\n", i);

}

if(inputFileContents[i] == ' '){

startOfWord = 1;

//word ended, so next iteration, store start of word

}

else {

lastWordSize++;

}

// printf("Last word size: %d\n", lastWordSize);

}

}

else if(S\_ISFIFO(inputFile.st\_mode)){

// printf("Stdin is not a regular file\n");

allStrings = (unsigned char\*\*)(malloc(sizeof(unsigned char\*)));

if(allStrings == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

while(alwaysTrue){

ssize\_t readResult = read(STDIN\_FILENO, inBuffer, 1);

if(readResult < 0){

fprintf(stderr, "Error when reading from stdin\n");

exit(1);

}

if(readResult == 0){ //reached end of file

break;

}

if(addNewString){

/\* start of new string \*/

numWords++;

/\* expand allStrings to hold another ptr \*/

allStrings = (unsigned char\*\*)(

realloc(allStrings, (numWords)\*sizeof(unsigned char\*)));

if(allStrings == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

/\* make new ptr for strings, allocate size for 1 char \*/

newString = (unsigned char\*)(malloc(sizeof(unsigned char)));

if(newString == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = inBuffer[0];

allStrings[numWords-1] = newString;

addNewString = 0;

newStringIndex++;

// printf("1Inserting :%c:\n", inBuffer[0]);

if(inBuffer[0] == ' '){

/\* space acts as "null byte" \*/

addNewString = 1;

newStringIndex = 0;

newString = NULL;

continue;

}

}

else if (!addNewString){

newString = (unsigned char\*)(

realloc(newString, (newStringIndex+1)\*sizeof(unsigned char)));

if(newString == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = inBuffer[0];

allStrings[numWords-1] = newString;

// printf("2Inserting :%c:\n", inBuffer[0]);

newStringIndex++;

if(inBuffer[0] == ' '){

/\* space acts as "null byte" \*/

addNewString = 1;

newStringIndex = 0;

newString = NULL;

}

}

// printf("Read in :%c:\n", inBuffer[0]);

}

if(numWords != 0 && newStringIndex != 0

&& allStrings[numWords-1][newStringIndex-1] != ' '){

/\* last char is not space \*/

newString = (unsigned char\*)(

realloc(newString, (newStringIndex+1)\*sizeof(unsigned char)));

if(newString == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

newString[newStringIndex] = ' ';

allStrings[numWords-1] = newString;

}

}

lastWordSize = lastWordSize - addedSpaceAtEnd;

// printf("REV last word size: %d\n", lastWordSize);

/\* Read in any bytes that have been appended to file \*/

unsigned char\* additionalContent=NULL;

int additionalSize = 0;

int addedSpaceAtEnd2 = 0;

int trueFileSize = fileSize - addedSpaceAtEnd;

//if empty file, then fileSize = addedSpaceAtEnd = 0

unsigned char tempBuffer[1];

int additionalSizeNEW = 0;

while(alwaysTrue){

ssize\_t readResult = read(STDIN\_FILENO, tempBuffer, 1);

if(readResult < 0){

fprintf(stderr, "Error in reading in data from file\n");

exit(1);

}

if(readResult == 0){

break;

}

additionalSizeNEW++;

}

lseek(STDIN\_FILENO, trueFileSize - lastWordSize, SEEK\_SET);

//move file pointer to start of last word in old file

if(additionalSizeNEW > 0){ //file size has increased

additionalSize = additionalSizeNEW + lastWordSize;

//subtract old filesize

if(!zeroSizeInitialFile){

numWords--; //roll back 1 word

wordIndex--;

}

fileSize = inputFile.st\_size; //then update filesize

// printf("FILE GREW, size is now %ld, increase of %d\n", inputFile.st\_size, (int)additionalSize - lastWordSize);

additionalContent = (unsigned char\*)(

malloc(additionalSize\*sizeof(unsigned char)));

// inputFileContents = (char\*)(realloc(inputFileContents, fileSize\*sizeof(unsigned char)));

// this line causes issues because reallocating invalidates all the old ptrs

if(additionalContent == NULL){

fprintf(stderr,

"Error in allocating memory to hold input file\n");

exit(1);

}

if(read(STDIN\_FILENO, additionalContent, additionalSize) < 0){

//read in additional bytes

fprintf(stderr, "Error when reading in from file\n");

}

for(int i = 0; i<additionalSize; i++){

if(additionalContent[i] == ' '){

numWords++;

}

}

if(additionalContent[additionalSize-1] != ' '){

// printf("ADD: Adding trailing space\n");

numWords++; //missing trailing space

addedSpaceAtEnd2 = 1;

additionalSize = additionalSize + 1;

additionalContent = (unsigned char\*)(

realloc(additionalContent, additionalSize\*sizeof(char)));

additionalContent[additionalSize-1] = ' ';

}

if(allStrings != NULL){

allStrings = (unsigned char\*\*)(

realloc(allStrings, numWords\*sizeof(unsigned char\*)));

}

else{

allStrings = (unsigned char\*\*)(

malloc(numWords\*sizeof(unsigned char\*)));

}

if(allStrings == NULL){

fprintf(stderr,

"Error in allocating memory to hold arr of strings\n");

exit(1);

}

// printf("ADD: File size is now %d\n", fileSize);

for(int i = 0; i<additionalSize; i++){

// printf("ADD: Analyzing :%c:\n", additionalContent[i]);

if(startOfWord){

allStrings[wordIndex] = &additionalContent[i];

wordIndex++;

startOfWord = 0;

// printf("ADD: Start of word at %d\n", i);

}

if(additionalContent[i] == ' '){

startOfWord = 1; //word ended, so next iteration, store start of word

}

}

if(addedSpaceAtEnd2){

// printf("2Added space at end\n");

}

}

// printf("FINAL num of words: %d\n", numWords);

if(allStrings != NULL){

qsort(allStrings, numWords, sizeof(unsigned char\*), compare);

printAllStrings(allStrings, numWords);

}

/\* clean up allocated memory \*/

if(S\_ISFIFO(inputFile.st\_mode)){

for(int i = 0; i<numWords; i++){

free(allStrings[i]);

}

}

free(allStrings);

free(inputFileContents);

free(additionalContent);

exit(0);

}

**Makefile:**

CC=gcc

all: tr2b tr2u sfrobu sfrobufast sfrobfast

tr2b: tr2b.c

gcc -o tr2b -g -W -Wall -pedantic -std=c11 tr2b.c

tr2u: tr2u.c

gcc -o tr2u -g -W -Wall -pedantic -std=c11 tr2u.c

sfrobu: sfrobu.c

gcc -o sfrobu -fsanitize=address -fsanitize=undefined \

-static-libubsan -static-libasan -std=c11 -g -W -Wall -pedantic sfrobu.c

sfrobufast: sfrobu.c

gcc -o sfrobufast -std=c11 -W -Wall -pedantic sfrobu.c

sfrobfast: sfrob.c

gcc -o sfrobfast -std=c11 -W -Wall -pedantic sfrob.c

clean:

rm -f tr2b tr2u sfrobu sfrobufast

**Lab 6:**

1. I used the following commands:

vim simpgmp.c

(copy pasted the text from the website to the file)

gcc -o simpgmp -lgmp simpgmp.c

2.

./simpgmp 24 outputted 16,777,216

./simpgmp `./simpgmp 24`

The second command outputted a number that started with 1818 and ended with

7536, as expected.

3. Using ldd simpgmp gives:

linux-vdso.so.1 => (0x00007ffda1bf9000)

libgmp.so.10 => /lib64/libgmp.so.10 (0x00007f804dd89000)

libc.so.6 => /lib64/libc.so.6 (0x00007f804d9bb000)

/lib64/ld-linux-x86-64.so.2 (0x00007f804e001000)

Thus, simpgmp is using the linux-vdso.so.1, libgmp.so.10, libc.so.6, and

ld-linux-x86-64.so.2 libraries.

4. Using the command strace ./simpgmp 24 gives the following output:

execve("./simpgmp", ["./simpgmp", "24"], [/\* 42 vars \*/]) = 0

brk(NULL) = 0x1b47000

mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f301395d000

access("/etc/ld.so.preload", R\_OK) = -1 ENOENT (No such file or directory)

open("/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=158981, ...}) = 0

mmap(NULL, 158981, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7f3013936000

close(3) = 0

open("/lib64/libgmp.so.10", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\200\304\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=495720, ...}) = 0

mmap(NULL, 2584736, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f30134c5000

mprotect(0x7f301353b000, 2093056, PROT\_NONE) = 0

mmap(0x7f301373a000, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x75000) = 0x7f301373a000

close(3) = 0

open("/lib64/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P&\2\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2156072, ...}) = 0

mmap(NULL, 3985888, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7f30130f7000

mprotect(0x7f30132ba000, 2097152, PROT\_NONE) = 0

mmap(0x7f30134ba000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c3000) = 0x7f30134ba000

mmap(0x7f30134c0000, 16864, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7f30134c0000

close(3) = 0

mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f3013935000

mmap(NULL, 8192, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f3013933000

arch\_prctl(ARCH\_SET\_FS, 0x7f3013933740) = 0

mprotect(0x7f30134ba000, 16384, PROT\_READ) = 0

mprotect(0x7f301373a000, 8192, PROT\_READ) = 0

mprotect(0x600000, 4096, PROT\_READ) = 0

mprotect(0x7f301395e000, 4096, PROT\_READ) = 0

munmap(0x7f3013936000, 158981) = 0

brk(NULL) = 0x1b47000

brk(0x1b68000) = 0x1b68000

brk(NULL) = 0x1b68000

fstat(1, {st\_mode=S\_IFCHR|0620, st\_rdev=makedev(136, 9), ...}) = 0

mmap(NULL, 4096, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_ANONYMOUS, -1, 0) = 0x7f301395c000

write(1, "16777216\n", 916777216

) = 9

exit\_group(0) = ?

+++ exited with 0 +++

Thus, I see that simpgmp makes the following system calls that relate to

dynamic linking: open, fstat, mmap, read, mprotect, close, and munmap.

Other system calls executed by the program but not directly related to linking

are brk, write, arch\_prctl, and access. Execve is used to start simpgmp.

mprotect, write, and exit\_group. There are 3 groups of function

calls that have to do with dynamic linking:

Group 1:

open("/etc/ld.so.cache", O\_RDONLY|O\_CLOEXEC) = 3

fstat(3, {st\_mode=S\_IFREG|0644, st\_size=158981, ...}) = 0

mmap(NULL, 158981, PROT\_READ, MAP\_PRIVATE, 3, 0) = 0x7fbb8da4c000

close(3) = 0

This serves to open the library ld.so.cache, find its file size, and then

allocate memory for it. It is closed when the program is done using it.

Group 2:

open("/lib64/libgmp.so.10", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\0\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0\200\304\0\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=495720, ...}) = 0

mmap(NULL, 2584736, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fbb8d5db000

mprotect(0x7fbb8d651000, 2093056, PROT\_NONE) = 0

mmap(0x7fbb8d850000, 12288, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x75000) = 0x7fbb8d850000

close(3) = 0

This serves to open the library libgmp.so.10, and then data is read in from

it, its file size is analyzed, memory is allocated for it, and it is

protected from it being overwritten access with mprotect. 832 bytes are read

in because that is the size of the ELF header of the file. It is closed when

the program is done using it.

Group 3:

open("/lib64/libc.so.6", O\_RDONLY|O\_CLOEXEC) = 3

read(3, "\177ELF\2\1\1\3\0\0\0\0\0\0\0\0\3\0>\0\1\0\0\0P&\2\0\0\0\0\0"..., 832) = 832

fstat(3, {st\_mode=S\_IFREG|0755, st\_size=2156072, ...}) = 0

mmap(NULL, 3985888, PROT\_READ|PROT\_EXEC, MAP\_PRIVATE|MAP\_DENYWRITE, 3, 0) = 0x7fbb8d20d000

mprotect(0x7fbb8d3d0000, 2097152, PROT\_NONE) = 0

mmap(0x7fbb8d5d0000, 24576, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_DENYWRITE, 3, 0x1c3000) = 0x7fbb8d5d0000

mmap(0x7fbb8d5d6000, 16864, PROT\_READ|PROT\_WRITE, MAP\_PRIVATE|MAP\_FIXED|MAP\_ANONYMOUS, -1, 0) = 0x7fbb8d5d6000

close(3) = 0

This serves to open the library libc.so.6, reads in data from it, allocates

memory, and protects the data from being accessed. It is very similar to what

was done with the libgmp.so.10 library. It is then closed when the program is

done using it.

5. I ran ls /usr/bin | awk '(NR-305087992)%251 == 0' and it outputted:

a2p

collateindex.pl

fc-pattern

gvfs-mime

kinfocenter

mkfifo

pango-querymodules-64

qdbuscpp2xml

stap-prep

wmf2x

6. I wrote a shell script to run ldd on each command:

#!/bin/bash

#program name: analyzeCmd

read -d '' allCommands

for c in $allCommands; do

echo 'ldd for' "$c"

path="/usr/bin/"

path+="$c"

ldd "$path"

done

Use the shell script in the following way:

ls /usr/bin | awk '(NR-305087992)%251 == 0' | ./analyzeCmd

I then analyzed the output and found that for a2p, fc-pattern, kinfocenter,

mkfifo, pango-querymodeuls64, qdbuscpp2xml, and wmf2x, ldd was able to work

normally. For collateindex.p1, gvfs-mime, and stap-prep, it returned the

error "not a dynamic executable", which occurs because these 3 programs are

shell scripts, not regular executables.

7. Using the following commands:

export LC\_ALL='C'

ls /usr/bin | awk '(NR-305087992)%251==0' | ./analyzeCmd | grep .so. | \

sed "s/[(].\*[)]//g" | sed "s/[=][>].\*//g" | sed 's/^[ \t]\*//g' | sort -u

I got the following list of libraries:

/lib64/ld-linux-x86-64.so.2...cut off the rest

Makefile:

OPTIMIZE = -O2

CC = gcc

CFLAGS = $(OPTIMIZE) -g3 -Wall -Wextra -march=native -mtune=native -mrdrnd

default: randall randmain randlibhw.so randlibsw.so

randall: randall.c

$(CC) $(CFLAGS) randall.c -o $@

# randmain.mk contains instructions for building

# randmain, randlibhw.so, and randlibsw.so.

-include randmain.mk

skeleton: dlskeleton.tgz

skeleton\_files = COPYING Makefile randall.c randcpuid.h randlib.h

dlskeleton.tgz: $(skeleton\_files)

tar -czf $@ --mode=a-w $(skeleton\_files)

submission: dlsubmission.tgz

submission\_files = lab.txt randmain.mk \

randcpuid.c randlibhw.c randlibsw.c randmain.c \

$(skeleton\_files)

dlsubmission.tgz: $(submission\_files)

tar -czf $@ $(submission\_files)

.PHONY: default clean skeleton submission

clean:

rm -f \*.o \*.so \*.so.\* \*.tgz randall randmain

Randmain.mk:

CC = gcc

CFLAGS = $(OPTIMIZE) -g3 -Wall -Wextra -march=native -mtune=native -mrdrnd

randmain: randmain.o randcpuid.o

$(CC) $(CFLAGS) -ldl -Wl,-rpath=$(PWD) -o randmain randmain.o randcpuid.o

randmain.o: randmain.c randcpuid.h

$(CC) $(CFLAGS) -c randmain.c

randcpuid.o: randcpuid.c randcpuid.h

$(CC) $(CFLAGS) -c randcpuid.c

randlibhw.o: randlibhw.c randlib.h

$(CC) $(CFLAGS) -fPIC -c randlibhw.c

randlibhw.so: randlibhw.o

$(CC) $(CFLAGS) -shared -o randlibhw.so randlibhw.o

randlibsw.o: randlibsw.c randlib.h

$(CC) $(CFLAGS) -fPIC -c randlibsw.c

randlibsw.so: randlibsw.o

$(CC) $(CFLAGS) -shared -o randlibsw.so randlibsw.o

Rancpuid.h:

extern \_Bool rdrand\_supported (void);

Randcpuid.c:

#include "randcpuid.h"

#include <cpuid.h>

/\* Implement function randcpuid.h, and copy over the cpuid struct, \*/

/\* as it's needed by the rdrand\_suppported function \*/

/\* Description of the current CPU. \*/

struct cpuid { unsigned eax, ebx, ecx, edx; };

/\* Return information about the CPU. See <http://wiki.osdev.org/CPUID>. \*/

static struct cpuid

cpuid (unsigned int leaf, unsigned int subleaf)

{

struct cpuid result;

asm ("cpuid"

: "=a" (result.eax), "=b" (result.ebx),

"=c" (result.ecx), "=d" (result.edx)

: "a" (leaf), "c" (subleaf));

return result;

}

/\* Return true if the CPU supports the RDRAND instruction. \*/

extern \_Bool

rdrand\_supported (void)

{

struct cpuid extended = cpuid (1, 0);

return (extended.ecx & bit\_RDRND) != 0;

}

Randlib.h:

extern unsigned long long rand64 (void);

randlibhw.c:

#include "randlib.h"

#include <immintrin.h>

/\* Return a random value, using hardware operations. \*/

extern unsigned long long

rand64 (void)

{

unsigned long long int x;

while (! \_rdrand64\_step (&x))

continue;

return x;

}

Randlibsw.c:

#include "randlib.h"

#include <stdio.h>

#include <stdlib.h>

/\* Software implementation. \*/

/\* Input stream containing random bytes. \*/

static FILE \*urandstream;

/\* Initialize the software rand64 implementation. \*/

\_\_attribute\_\_ ((constructor))

static void

software\_rand64\_init (void)

{

urandstream = fopen ("/dev/urandom", "r");

if (! urandstream)

abort ();

}

/\* Return a random value, using software operations. \*/

extern unsigned long long

rand64 (void)

{

unsigned long long int x;

if (fread (&x, sizeof x, 1, urandstream) != 1)

abort ();

return x;

}

/\* Finalize the software rand64 implementation. \*/

\_\_attribute\_\_ ((destructor))

static void

software\_rand64\_fini (void)

{

fclose (urandstream);

}

Randmain.c:

#include "randcpuid.h"

#include <errno.h>

#include <dlfcn.h>

#include <stdio.h>

#include <stdlib.h>

#include <stdbool.h>

static bool

writebytes (unsigned long long x, int nbytes)

{

int ndigits = nbytes \* 2;

do

{

if (putchar ("0123456789abcdef"[x & 0xf]) < 0)

return false;

x >>= 4;

ndigits--;

}

while (0 < ndigits);

return 0 <= putchar ('\n');

}

int

main (int argc, char \*\*argv)

{

/\* Check arguments. \*/

bool valid = false;

long long nbytes;

if (argc == 2)

{

char \*endptr;

errno = 0;

nbytes = strtoll (argv[1], &endptr, 10);

if (errno)

perror (argv[1]);

else

valid = !\*endptr && 0 <= nbytes;

}

if (!valid)

{

fprintf (stderr, "%s: usage: %s NBYTES\n", argv[0], argv[0]);

return 1;

}

/\* If there's no work to do, don't worry about which library to use. \*/

if (nbytes == 0)

return 0;

/\* Now that we know we have work to do, arrange to use the

appropriate library. \*/

unsigned long long (\*rand64) (void)

void \*libHandle;

if (rdrand\_supported ())

{

//hardware implementation exists so we load hw library

libHandle = dlopen("./randlibhw.so", RTLD\_LAZY);

if(!libHandle){

fprintf(stderr, "Error when opening library: %s\n", dlerror());

exit(1);

}

rand64 = dlsym(libHandle, "rand64");

if(dlerror() != NULL){

fprintf(stderr, "Error finding symbol\n");

exit(1);

}

}

else

{

libHandle = dlopen("./randlibsw.so", RTLD\_LAZY);

if(!libHandle){

fprintf(stderr, "Error when opening library: %s\n", dlerror());

exit(1);

}

rand64 = dlsym(libHandle, "rand64");

if(dlerror() != NULL){

fprintf(stderr, "Error finding symbol\n");

exit(1);

}

}

//no need to initialize cause that gets done when the library is loaded

int wordsize = sizeof rand64 ();

int output\_errno = 0;

do

{

unsigned long long x = rand64 ();

int outbytes = nbytes < wordsize ? nbytes : wordsize;

if (!writebytes (x, outbytes))

{

output\_errno = errno;

break;

}

nbytes -= outbytes;

}

while (0 < nbytes);

if (fclose (stdout) != 0)

output\_errno = errno;

if (output\_errno)

{

errno = output\_errno;

perror ("output");

return 1;

}

//no need to finalize b/c library does that when closing

if(dlclose(libHandle)){

fprintf(stderr, "Error when closing dynamic library\n");

exit(1);

}

return 0;

}

**Lab #7:**

1. I ran git clone https://git.savannah.gnu.org/git/diffutils.git, which was

taken from the GNU diff utiliies website.

2. I then entered the newly created diffutils folder and checked what changes

had been made:

cd diffutils

git log > git-log.txt

3. To generate a list of tags, I used the following command:

git tag > git-tags.txt

4. To find the commit's hash, I used:

cat git-log.txt | grep -C5 'maint: quote'

I found the hash was 62ca21c8c1a5aa3488589dcb191a4ef04ae9ed4f, so I used the

following command to generate a patch:

git format-patch -1 62ca21c8c1a5aa --stdout > quote-patch.txt

5. I checked out version 3.0 by using the following command:

git checkout tags/v3.0

6. I then made the patch using:

patch -p1 < quote-patch.txt

The following errors were returned:

can't find file to patch at input line 62

...

patching file README

Hunk #1 FAILED at 9.

Hunk #3 FAILED at 60.

2 out of 3 hunks FAILED -- saving rejects to file README.rej

patching file README-hacking

Hunk #1 FAILED at 19.

1 out of 1 hunk FAILED -- saving rejects to file README-hacking.rej

patching file TODO

patching file cfg.mk

Hunk #1 FAILED at 29.

1 out of 1 hunk FAILED -- saving rejects to file cfg.mk.rej

patching file doc/diagmeet.note

can't find file to patch at input line 249

...

can't find file to patch at input line 262

...

patching file src/diff.c

...

Hunk #10 FAILED at 876.

Hunk #11 FAILED at 906.

Hunk #12 FAILED at 942.

...

3 out of 14 hunks FAILED -- saving rejects to file src/diff.c.rej

patching file src/diff3.c

...

Hunk #6 FAILED at 433.

...

1 out of 9 hunks FAILED -- saving rejects to file src/diff3.c.rej

7. git status returns the following output:

HEAD detached at v3.0

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: NEWS

modified: README

modified: TODO

modified: doc/diagmeet.note

modified: ms/config.bat

modified: ms/config.site

modified: po/en.po

modified: src/analyze.c

modified: src/cmp.c

modified: src/context.c

modified: src/diff.c

modified: src/diff.h

modified: src/diff3.c

modified: src/dir.c

modified: src/ifdef.c

modified: src/io.c

modified: src/sdiff.c

modified: src/side.c

modified: src/system.h

modified: src/util.c

modified: tests/help-version

Untracked files:

(use "git add <file>..." to include in what will be committed)

NEWS.orig

README-hacking.orig

README-hacking.rej

README.orig

README.rej

cfg.mk.orig

cfg.mk.rej

git-log.txt

git-tags.txt

ms/config.site.orig

quote-patch.txt

src/cmp.c.orig

src/context.c.orig

src/diff.c.orig

src/diff.c.rej

src/diff.h.orig

src/diff3.c.orig

src/diff3.c.rej

src/dir.c.orig

src/sdiff.c.orig

src/system.h.orig

src/util.c.orig

tests/help-version.orig

no changes added to commit (use "git add" and/or "git commit -a")

8. Using emacs to open a file (such as NEWS), I ran the following commands to

learn about what they do:

C-x v = (vc-diff): This creates another window that compares the current file

to the same file that I started with. It essentially works like diff. More

precisely, the online documentation states that it compares the work files

in the current version control fileset to the starting versions.

C-x v u (vc-revert): This gives me the option of reverting all my changes so

that the file returns to its original version. More precisely, the online

documentation says it reverts the work file in the current version control

fileset to the last revision.

While in the \*vc-diff\* buffer, C-h m displays additional commands that can

be used.

C-c C-a (diff-apply-hunk): This applys a certain diff-hunk to the target file.

C-u can be added to revert the hunk (aka undo the change).

C-c C-c (diff-goto-source): This goes to the specific line in the new source

file that corresponds to a hunk. A prefix can be added to jump to the source

line in the old file instead.

9. I reverted the changes by opening the file, typing C-x v u, and then yes

when asked if I wanted to discard changes. I did this for NEWS, README, TODO,

doc/diagmeet.note, ms/config.bat, ms/config.site, po/en.po, src/system.h,

and tests/help-version.

To undo the changes to any non-character string constants, I opened each

.c file, used C-x v = to examine where changes occurred, and if they needed

to be reverted, used C-u C-c C-a.

Completley reverted:

src/analyze.c

src/context.c

src/ifdef.c

src/io.c

src/side.c

Changes remain:

src/cmp.c

src/diff.c

src/diff3.c

src/dir.c

src/sdiff.c

src/util.c

10. THe files that contained rejected changes are as follows:

diff3.c.rej

diff.c.rej

I opened these files using emacs, and then opened their corresponding C file

in the same frame using C-x 4 f [diff3.c or diff.c], and then made the

appropriate changes from ` to ' for any constant string literals.

11. I removed all the untracked files by moving them to a different directory

using mv, and then used git clean -n to see which files would be removed:

Would remove NEWS.orig

Would remove README-hacking.orig

Would remove README-hacking.rej

Would remove README.orig

Would remove README.rej

Would remove cfg.mk.orig

Would remove cfg.mk.rej

Would remove git-log.txt

Would remove git-tags.txt

Would remove ms/config.site.orig

Would remove quote-patch.txt

Would remove src/cmp.c.orig

Would remove src/context.c.orig

Would remove src/diff.c.orig

Would remove src/diff.c.rej

Would remove src/diff.h.orig

Would remove src/diff3.c.orig

Would remove src/diff3.c.rej

Would remove src/dir.c.orig

Would remove src/sdiff.c.orig

Would remove src/system.h.orig

Would remove src/util.c.orig

Would remove tests/help-version.orig

These are all the unmodified filess, or files that were created during this

lab, so I ran git clean -f to remove them.

12. Running git status gives the following output:

HEAD detached at v3.0

Changes not staged for commit:

(use "git add <file>..." to update what will be committed)

(use "git checkout -- <file>..." to discard changes in working directory)

modified: src/cmp.c

modified: src/diff.c

modified: src/diff3.c

modified: src/dir.c

modified: src/sdiff.c

modified: src/util.c

I then used the command git diff > quote-3.0-patch.txt to create a new patch file.

13. I first read the README-hacking file to figure out how to build the

modified version of Diffutils. IT says to run the following commands:

./bootstrap

./configure

make

make check

The first three commands ran fine, but make check returned a large amount of errors:

...

make[3]: Entering directory '/w/home.23/ma/ugrad/danning/cslab/diffutils/tests'

/bin/sh: /bin/sh: cannot execute binary file

Makefile:1309: recipe for target 'basic.log' failed

make[3]: \*\*\* [basic.log] Error 126

make[3]: Leaving directory '/w/home.23/ma/ugrad/danning/cslab/diffutils/tests'

Makefile:1288: recipe for target 'check-TESTS' failed

make[2]: \*\*\* [check-TESTS] Error 2

make[2]: Leaving directory '/w/home.23/ma/ugrad/danning/cslab/diffutils/tests'

Makefile:1403: recipe for target 'check-am' failed

make[1]: \*\*\* [check-am] Error 2

make[1]: Leaving directory '/w/home.23/ma/ugrad/danning/cslab/diffutils/tests'

Makefile:1016: recipe for target 'check-recursive' failed

make: \*\*\* [check-recursive] Error 1

I tried doing the same ./bootstrap, ./configure, make, and make check on a

clean (unmodified) v3.0 branch of Diffutils, and similar errors were returned.

14. I copied all the modified source code into a folder called diffutils-3.0-patch:

cd ..

mkdir diffutils-3.0-patch

cp -r diffutils/src/\*.c diffutils-3.0-patch

To get a fresh version of the v3.0 branch of Diffutils, I ran the following commands:

mkdir diffutils-fresh

cd diffutils-fresh

git clone https://git.savannah.gnu.org/git/diffutils.git

git checkout tags/v3.0

cd ..

mkdir diffutils-3.0

cp -r diffutils-fresh/diffutils/src/\*.c diffutils-3.0

Then, to compare the two sets of source codes, I used the following command:

diffutils/src/diff -pru diffutils-3.0 diffutils-3.0-patch > quote-3.0-test.txt

15. I ran the following command to compare quote-3.0-test.txt and quote-3.0-patch.txt:

diff quote-3.0-test.txt diffutils/quote-3.0-patch.txt > differences.txt

Examining differences.txt shows that basically the only changes are from ` to ', as expected.

Hw7.txt:

Danning Yu

305087992

Throughout this lab, I used lnxsrv05.

1. I used the command emacs hw7.txt.

2. To get a new branch called quote with v3.0 of Diffutils, I used:

cd diffutils

git checkout v3.0 -b quote

This gave the following output:

src/cmp.c

src/diff.c

src/diff3.c

src/dir.c

src/sdiff.c

src/util.c

Switched to a new branch 'quote'

3. To apply the patch, I used:

patch -p1 < quote-3.0-patch.txt

4. The emacs command C-x 4 a (add-change-log-entry-other-window) allows for

a new entry to be added to the change log of the file that is currently

being edited.

5. I opened the patch using emacs quote-3.0-patch.txt, and then wrote the

following entry:

2019-11-16 Danning Liu Yu <danning@lnxsrv05.seas.ucla.edu>

\* quote-3.0-patch.txt: This patch is in response to a recent change in

GNU coding standards, which now suggest quoting 'like this' or "like

this", instead of `like this' or ``like this''. This change only

applies to character string constants in .c src files.

Following files were changed:

src/cmp.c

src/diff.c

src/diff3.c

src/dir.c

src/sdiff.c

src/util.c

I then saved the file (C-x C-s) and exited (C-x C-c).

6. I moved the patch out of the diffutils folder and then commited using the

following commands:

mv quote-3.0-patch.txt ../quote-3.0-patch.txt

git add .

git commit -F ChangeLog

7. I used the following command to generated a formatted patch:

git format-patch -1 --stdout > formatted-patch.txt

I checked the formatted patch file and looks very similar to the file

quote-3.0-patch.txt, as expected.

8. Partner: Janis Chen, UID: 105096822

I made a new folder, cloned the respository into the folder, and then switched

it to the v3.0 commit on a new branch called partner:

git clone https://git.savannah.gnu.org/git/diffutils.git

cd diffutils

git checkout tags/v3.0

git checkout v3.0 -b partner

git am formatted-patch-JC.txt

./bootstrap

./configure

make

make check

It passed some test cases, but the ones that it failed were the same as ones

that failed in the version of Diffutils that I used to generate this patch.

I manually opened some of the .c files and verified that the ` characters in

constant string literals had been changed to '.

9. I verified the changelog entry was there by running the following commands:

make distdir

This created a new directory called diffutils-3.0.1-6dd9, and within that

contains the ChangeLog with my partner's commit message.

cd diffutils-3.0.1-6dd9

emacs ChangeLog

Indeed, the commit message from my partner's patch was there.

10. I fist installed Xming onto my computer, and then enabled X11 forwarding

in PuTTY. Then, I ran the following commands:

cd ~eggert/src/gnu/emacs

gitk

This caused a new window to open, containing a GUI representation of the git

commit history for the emacs repository. The newest merge not newer than

2015-01-25 is merging the 'master' branch into xwidget. The top left window

shows the two branches and then getting merged together, hte window next to it

shows the author of the operation, and the window to the rigt of that shows

the date this operation occurred. In the bottom left window, it contains the

commit/merge message, and the to the right of that window is a window for

comments. The SHA1 of the merge is also displayed, as well as a simple GUI

for finding specific commits.

**Lab 8:**

**Log.txt (lab):**

Setting up the BeagleBone:

To setup the BeagleBone, I carried out the following steps, following the

instructions outlined on Piazza:

1. Connected BeagleBone to my computer using the provided uSB cable. Opened it

as an external storage device. Opened the START.htm file and installed the

appropriate drivers.

2. ssh'd into the device using PuTTY, with the host being 192.168.7.2, on port

22. Logged in as the root user.

3. Connected to the wifi:

connmanctl

enable wifi

scan wifi

services (this lists all the wifi access points found during the scan)

agent on

connect [string next to wifi name]

quit

4. Install updated packages:

sudo apt-get update

sudo apt-get install xauth

sudo apt-get install xvfb

sudo apt-get install firefox-esr-l10n-en-gb

5. Test X11 forwarding: I typed "firefox" into the terminal and indeed, an

XMing window popped up with firefox on it. However, there were a large number

of errors that popped up, of this form:

(/usr/lib/firefox-esr/firefox-esr:1613): GLib-GObject-CRITICAL \*\*:

g\_object\_unref: assertion 'object->ref\_count > 0' failed

However, these errors did not seem to really affect the functionality of

Firefox.

I was able to complete all of these steps, thus setting up my BeagleBone.

--------------------

Setting up the server:

To generate a public-private key pair:

ssh-keygen

When prompted for the directory, I pressed enter to save the key to the

default directory (/root.ssh/id\_rsa). Then, I used "123456" for the passphrase.

The following output resulted:

Your identification has been saved in /root/.ssh/id\_rsa.

Your public key has been saved in /root/.ssh/id\_rsa.pub.

The key fingerprint is:

4a:44:f0:d5:3f:ab:50:1c:ce:ad:bd:e6:16:b8:23:2e root@beaglebone

The key's randomart image is:

+---[RSA 2048]----+

| ... .. |

| o . o |

| o + + |

| . = + |

| . S. + o |

| . .. o + |

| . . o o |

| E . + + |

| o.. =. |

+-----------------+

I then created a new user with username "user1" and password "abcdef":

cd /home

sudo useradd -d /home/user1 -m user1

sudo passwd user1

Enter new UNIX password: abcdef

Retype new UNIX password: abcdef

passwd: password updated successfully

I then created a .ssh directory and let user1 be its owner and changed the

directory's permissions:

cd /home/user1

sudo mkdir .ssh

sudo chown -R user1 .ssh

sudo chmod 700 .ssh

Finally, I got the IP address of the BeagleBone so that I could give it to

anyone that wanted to connect as a client:

hostname -I

------------------------------

Setting up the client:

Generate key:

ssh-keygen

It asked if I wanted to override the previously generated key. I typed "y

for yes.

ssh-copy-id -i joshualiu97@192.168.137.211

It then asked me if I trusted the server, as I had never logged in before.

I responded with "yes", and then I typed in the password for the joshualiu97

account, which my partner created.

Then, I started ssh-agent:

eval $(ssh-agent)

It outputted the following line:

Agent pid 2664

Then I added my private key to the authentication agent:

ssh-add

Then it said the key was added, as expected.

Then I logged in again:

ssh joshualiu97@192.168.137.211

It did not ask me for a password this time because ssh-agent saved it already.

I then created a file:

vim testfile.txt

My partner was able to see the file on his terminal.

To test X11 forwarding, I used:

ssh -X joshualiu97@192.168.137.211

Then I tried to open Firefox:

firefox

An XMing window with Firefox popped up, as expected.

**Hw8.txt:**

To generate a key-pair, I used gpg2:

gpg2 --gen-key

It gave the following output:

gpg: key 34D550A5 marked as ultimately trusted

public and secret key created and signed.

gpg: checking the trustdb

gpg: 3 marginal(s) needed, 1 complete(s) needed, PGP trust model

gpg: depth: 0 valid: 1 signed: 0 trust: 0-, 0q, 0n, 0m, 0f, 1u

pub 2048R/34D550A5 2019-11-23

Key fingerprint = 2E71 6F4D F1AC 6541 6A13 7E35 D6D9 2CB2 34D5 50A5

uid Danning Yu (Key 1) <danningyu@ucla.edu>

sub 2048R/DE55498F 2019-11-23

To export the key:

gpg2 --armor --export danningyu@ucla.edu > hw-pubkey.asc

To transfer the eeprom file from BeagleBone to the Linux server (ssh'd into

BeagleBone already via PuTTY).

scp -r /sys/bus/i2c/devices/0-0050/eeprom danning@lnxsrv07.seas.ucla.edu:eeprom

Generating deatached signature for eeprom using the key previously generated:

gpg2 --clearsign eeprom

This command returned the following messages:

2048-bit RSA key, ID 34D550A5, created 2019-11-23

gpg: input line longer than 19995 characters

Then I ran the follow command to create the deatched sign:

gpg2 --detach-sign eeprom

--------------------

Homework Questions

1. If other teams had been monitoring all the bytes that went across the

network, the resulting transmissions are still secure because we always used

SSH to log in to each other's BeagleBone hosts. SSH encrypts transmissions so

they can only be decrypted by the server and client. Anyone monitoring the

network would only be able to see the encrypted data being sent over the

network, and they cannot decrypt it because they do not have private key.

If case (1) occurred, the connection would still be secure because when I set

the connection with my partner, I made sure to set up key-based authentication

and then add my private to the authentication agent, which meant that when I

tried to log in to this server in the future, I would not have to type in my

password again. Thus, if my keyboard was tapped after all of this was

completed, the network would still be secure because they would never obtain

my password.

If case (2) occurred, the connection would not be secure because they could

access the private keys stored on the USB device, thus allowing them to

decrypt anything encryped with my public key. The .ssh folder on the USB

would contain all public and private keys, and they would have access to all

of them, thus allowing them to decrypt messages sent on the network.

2. The gpg2 --verify command doesn't really verify that I created the file

because when creating the key, I could have put in anyone's name and email,

thus linking it to that key. All gpg2 --verify does is compare the file

matches with the detached signature. Thus, if someone intercepts the file,

modifies it, and then generates a new detached signature using their own key,

running gpg2 --verify on that file will show the person who intercepted the

file as being the creator of the file (or my name if they choose to

impersonate me). To fix this issue, we need to use a 3rd party, called a

certificate authority, to certify that a certain person owns a particular

public key. Then, whoever receives the file can use the appropriate public

key (mine) to verify the file was created by me. If it was not, or it was

tampered with in transit, then using my public key with gpg2 --verify will

result in the detached signature and target file not matching, as desired.

**Lab 9:**

First I entered the emacs repository for CS 35L:

cd ~eggert/src/gnu/emacs-CS-35L

1. Using the following commands:

man du

du -a | sort -n -r | head -n 2

find . -type f -exec du -a {} + | sort -n -r | head -n 10

Output of 2nd command (after man du):

520828 .

358316 ./.git

The second command (after man du) gives that the entire directory is 520828 bytes and that the

./.git directory is 358316 bytes. Thus, the working files take up 520828 - 358316 = 162512 bytes,

and the Git repository takes up 358316 bytes.

The third command gives the following output:

318152 ./.git/objects/pack/pack-24e56b1749b9320c560213d9045fa6cee42b4174.pack

23876 ./.git/objects/pack/pack-24e56b1749b9320c560213d9045fa6cee42b4174.idx

7888 ./.git/objects/info/commit-graph

6836 ./.git/gitk.cache

6736 ./test/manual/BidiCharacterTest.txt

4400 ./leim/SKK-DIC/SKK-JISYO.L

2508 ./admin/unidata/NormalizationTest.txt

2280 ./ChangeLog.3

1760 ./admin/unidata/UnicodeData.txt

1696 ./admin/charsets/mapfiles/cns2ucsdkw.txt

The pack files are large because they contain past versions of the emacs project, and because

the emacs project has been going on for a long time (this repository's history dates back to the

1990s), there are lots of old files. Also, there are most likely large binary blob files that take

up a lot of space, and these files cannot be efficiently compressed.

2. I used the following commands:

git branch | wc -l

git branch -r | wc -l

The first command gave the number of local branches to be 176, and the second command gave the

number of remote branches to be 177.

3. Using the following command:

git remote -v

This displays one remote repository, called origin, located at

https://git.savannah.gnu.org/git/emacs.git. It is used for both fetching and pushing.

4. I used the following command:

git branch --sort=-committerdate | head -n 10

This gave the following output:

master

scratch/joaot/make-completion-at-point-function

feature/windows-with-utils

scratch/completion-api

scratch/a-modest-completion-redesign-proposal

scratch/fido-mode

feature/gnus-select2

feature/extend\_face\_id

scratch/jit-lock-antiblink-cleaned-up

emacs-26

Each branch should also be prepended with refs/heads/ for the full branch name.

5. Using the following command gives that there are 139583 commits on the master branch:

git rev-list --count master

6. Using the following command gives the total number of commits across all branches to be 143910:

git rev-list --all --count

Dividing the answer to #5 by this number gives 139583/143910, or approximately 96.99%.

7. Using the following command gives the desired result, where the number in the first column is

the number of commits and the entry in the second column is the author.

git shortlog -sn master --since=2013.01.01 | head -n 10

3691 Eli Zaretskii

3647 Glenn Morris

3605 Paul Eggert

1806 Lars Ingebrigtsen

1784 Stefan Monnier

1571 Michael Albinus

619 Dmitry Gutov

576 Noam Postavsky

471 Alan Mackenzie

469 Juri Linkov

8. Using gitk, I looked up the commits listed in the assignnment spec to create the following

commit graph. Some branches and/or commits not shown for clarity.

(Note: The graph is made up of 10 lines of text, each line having up to 150 characters)

[graph excluded]

Basically, what is happening is that after a change to "Cope better with C++..." (commit 4ea37c),

this change is merged into the master branch and emacs is incremented to version

27.0.50 (commit 977cd), and then an emacs 26 release branch is started (commit 625cee). On this

branch, changes are made such as adding a lisp variable (5490ccc), and then this change is

eventually merged into the master branch. Then, a new branch is created where eventually, a change

is made to doc/emacs/display... (commit 820739), and then from this commit, a branch is created to

fix an undecorated frame resizing issue (commit 004e3e), then another branch that eventually has a

change in test/lisp/tramp-tests.el (commit 49cd561). This final branch is merged into the master

branch (commit abcb2e), and then after a couple more changes are made on the branch that had

commit 49cd561, it is again merged into the master branch (commit 98ac35e). Essentially, what is

happening is emacs was incremented to version 27, and then changes/updates are being made on the

emacs 26 release branch before being merged with the main branch (representing emacs version 27).