Homework #1

Advanced Programming in the UNIX Environment

Due: ~~March 21~~March 29, 2022

Read carefully before you implement and submit your homework.

Implement a 'lsof'-like program

In this homework, you have to implement the 'lsof' tool by yourself. 'lsof' is a tool to list open files. It can be used to list all the files opened by processes running in the syste output of your homework is required to follow the spec *strictly*. The TAs will use the 'diff' tool to compare your output directly against our prepared sample test data.

Program Arguments

Your program should work without any arguments. In the meantime, your program has to handle the following arguments properly:

- `-c REGEX` : a regular expression (REGEX) filter for filtering command line. For example `-c sh` would match `bash` , `zsh` , and `share` .
- `-t TYPE` : a TYPE filter. Valid TYPE includes `REG` , `CHR` , `DIR` , `FIFO` , `SOCK` , and `unknown` . TYPEs other than the listed should be considered invalid. For invalid typ program has to print out an error message `Invalid TYPE option.` in a single line and terminate your program.
- `-f REGEX` : a regular expression (REGEX) filter for filtering filenames.

A sample output from this homework is demonstrated as follows:

```
$ ./hw1 | head -n 20
COMMAND         PID             USER            FD              TYPE            NODE            NAME
systemd         1               root            cwd             unknown                         /proc/1/cwd (Permission denied)
systemd         1               root            rtd             unknown                         /proc/1/root (Permission denied)
systemd         1               root            txt             unknown                         /proc/1/exe (Permission denied)
systemd         1               root            NOFD                                            /proc/1/fd (Permission denied)
kthreadd                2               root            cwd             unknown                         /proc/2/cwd (Permission denied)
kthreadd                2               root            rtd             unknown                         /proc/2/root (Permission denied)
kthreadd                2               root            txt             unknown                         /proc/2/exe (Permission denied)
kthreadd                2               root            NOFD                                            /proc/2/fd (Permission denied)
rcu_gp          3               root            cwd             unknown                         /proc/3/cwd (Permission denied)
rcu_gp          3               root            rtd             unknown                         /proc/3/root (Permission denied)
rcu_gp          3               root            txt             unknown                         /proc/3/exe (Permission denied)
rcu_gp          3               root            NOFD                                            /proc/3/fd (Permission denied)
rcu_par_gp              4               root            cwd             unknown                         /proc/4/cwd (Permission denied)
rcu_par_gp              4               root            rtd             unknown                         /proc/4/root (Permission denied)
rcu_par_gp              4               root            txt             unknown                         /proc/4/exe (Permission denied)
rcu_par_gp              4               root            NOFD                                            /proc/4/fd (Permission denied)
kworker/0:0H-events_highpri             6               root            cwd             unknown                         /proc/6/cwd (Perm
kworker/0:0H-events_highpri             6               root            rtd             unknown                         /proc/6/root (Per
kworker/0:0H-events_highpri             6               root            txt             unknown                         /proc/6/exe (Perm
...
```

The detailed spec of this homework is introduced as follows. Your program has to output the following fields (columns) for each file opened by a running process. Each line p the information for a single file. The required fields include `COMMAND` , `PID` , `USERM` , `FD` , `TYPE` , `NODE` , and `NAME` . The meaning of each field (column) is explained below.

- `COMMAND` :
  - The **executable filename** of a running process.
  - DO NOT show arguments.
- `PID` :
  - Process id of a running process.
  - Only need to handle opened files in process level (check `/proc/[pid]` . No need to handle opened files at thread level (that would be in `/proc/[pid]/task/[`
- `USER` :
  - The username who runs the process.
  - Please show `username` instead of UID.
- `FD` : The file descriptor. The value shown in the `FD` field can be one of the following cases.
  - `cwd` : The current working directory, which can be read from `/proc/[pid]/cwd` .
  - `rtd` : root directory, which can be read from `/proc/[pid]/root` .
  - `txt` : program file of this process, can be read from `/proc/[pid]/exe` .
  - `mem` : memory mapping information, which can be read from `/proc/[pid]/maps` .
    - If `/proc/<pid>/maps` is not accessible, you don't need to show any information about mapped files.
    - A memory-mapped file may have multiple segments or be mapped multiple times. You only need to output the first one for duplicated files, i.e., files havin same i-node or filename.
    - You don't need to handle memory segments that do not associate with a file. For example, [heap] or anonymously mapped memory segments. Those mer segments should have an i-node number of zero.
  - `DEL` : indicate a memory-mapped file has been deleted. You should show this value if there is a (deleted) mark right after the filename in memory maps.
  - `[0-9]+[rwu]` : file descriptor and opened mode.
    - The numbers show the file descriptor number of the opened file.
    - The mode "r" means the file is opened for reading.
    - The mode "w" means the file is opened for writing.
    - The mode "u" means the file is opened for reading and writing.
  - `NOFD` : if `/proc/[pid]/fd` is not accessible. In this case, the values for `TYPE` and `NODE` fields can be left empty.
- `TYPE` : The type of the opened file. The value shown in `TYPE` can be one of the following cases.

    - `DIR` : a directory. `cwd` and `rtd` are also classified as this type.
    - `REG` : a regular file
    - `CHR` : a character special file, for example

      ```
      crw-rw-rw- 1 root root 1, 3 Mar 17 17:31 /dev/null
      ```

    - `FIFO` : a pipe, for example
        - A link to a pipe, e.g.,

          ```
          lr-x------ 1 root root 64 Mar 17 19:55 5 -> 'pipe:[138394]'
          ```

        - A file with `p` type (FIFO)

          ```
          prw------- 1 root root 0 Mar 17 19:54 /run/systemd/inhibit/11.ref
          ```

    - `SOCK` : a socket, for example

      ```
      lrwx------ 1 root root 64 Mar 17 19:55 1 -> 'socket:[136975]'
      ```

    - `unknown` : Any other unlisted types. Alternatively, if a file has been deleted or is not accessible (e.g., permission denied), this column can show `unknown` .
- `NODE` :
    - The i-node number of the file
    - It can be blank or empty if and only if `/proc/[pid]/fd` is not accessible.
- `NAME` :
    - Show the opened filename if it is a typical file or directory.
    - Show `pipe:[node number]` if it is a symbolic file to a pipe, e.g.,

      ```
      l-wx------ 1 ta ta 64 三 8 02:11 91 -> 'pipe:[2669735]'
      ```

    - Show `socket:[node number]` if it is a symbolic file to a socket, e.g.,

      ```
      lrwx------ 1 ta ta 64 三 8 02:11 51 -> 'socket:[2669792]'
      ```

    - Append `(Permission denied)` if the access to `/proc/pid/fd` or `/proc/pid/(cwd|root|exe)` is failed due to permission denied.
    - If the filename you read from /proc file system contains a `(deleted)` , please remove it from the filename before you print it out.

Additional Notes on REGEX

If you plan to test REGEX feature with the `lsof` package that comes with Linux distributions, you should run it with the option `-c /REGEX/` .

For students who programming with C++, consider working with `regex_search()` instead of `regex_match()` . Please work with `regcomp()` and `regexec()` for student
are programming with C to implement this feature.

Grading

- We will test your program in Ubuntu 20.04 with some simple test cases. You may not have to consider peaky test cases.
- All continuous spaces and tabs in the output are considered as a single space. You may not have to worry about the differences between space characters in program
- For each test case, if more than 90% of your output lines are equivalent to our sample output, you will get full scores for that test case.
- If your program crashes in a test case, you will only get part or no scores. It is case-by-case.
- All the required program options and arguments can be passed to the program simultaneously. But we will not run it with ambiguous or redundant program options like `./hw1 -c A -c B` .
- Because a running process in the system could be terminated at any time, your program may encounter a race condition that an earlier check is successful for `/proc/`
  but latter accesses to files in `/proc/<pid>` directory are failed. Your program should be able to handle cases like this properly.
- We have provided a sample implementation with three test cases. Please access our online sandbox (https://up.zoolab.org/service/) (tested only with Chrome, Firefox,
  Edge; Safari is not supported). You can find everything in the directory `~/uphw1` . Note that you can only access the service within campus networks or via a valid VPN
  network.

Homework Submission

We will compile your homework by simply typing 'make' in your homework directory. You have to ensure your Makefile produces the executable hw1. Please ensure your Mal
works and the output executable name is correct before submitting your homework.

Please pack your C/C++ code and Makefile into a **zip** archive. The directory structure should follow the below illustration. The *id* is your student id. Please note that you don'
enclose your id with the braces.

```
{id}_hw1.zip
└── {id}_hw1/
        ├── Makefile
        ├── hw1.cpp
        └── (any other c/c++ files if needed)
```

You have to submit your homework via the E3 system. Scores will be graded based on the completeness of your implementation.

Remarks

- Please implement your homework in C or C++.
- Using any non-standard libraries and any external binaries (e.g., via system()) are not allowed.
- No copycats. Please do not use codes from others (even open-source projects).
- We will test your program in **Ubuntu 20.04 LTS** Linux with the default gcc version (9.3.0).

More Outputs

Your program has to order the output lines by performing a numeric sort against process ID (PIDs). We will test your program with and without root permission. If an operatior
not have sufficient permission to perform, you have to print out `Permission denied` message.

Run the command without root permission

```
$ ./hw1| head -n 20
COMMAND         PID             USER            FD              TYPE            NODE            NAME
systemd         1               root            cwd             unknown                         /proc/1/cwd (Permission denied)
systemd         1               root            rtd             unknown                         /proc/1/root (Permission denied)
systemd         1               root            txt             unknown                         /proc/1/exe (Permission denied)
systemd         1               root            NOFD                                            /proc/1/fd (Permission denied)
kthreadd                2               root            cwd             unknown                         /proc/2/cwd (Permission denied)
kthreadd                2               root            rtd             unknown                         /proc/2/root (Permission denied)
kthreadd                2               root            txt             unknown                         /proc/2/exe (Permission denied)
kthreadd                2               root            NOFD                                            /proc/2/fd (Permission denied)
kworker/0:0H            4               root            cwd             unknown                         /proc/4/cwd (Permission denied)
kworker/0:0H            4               root            rtd             unknown                         /proc/4/root (Permission denied)
kworker/0:0H            4               root            txt             unknown                         /proc/4/exe (Permission denied)
kworker/0:0H            4               root            NOFD                                            /proc/4/fd (Permission denied)
mm_percpu_wq            6               root            cwd             unknown                         /proc/6/cwd (Permission denied)
mm_percpu_wq            6               root            rtd             unknown                         /proc/6/root (Permission denied)
mm_percpu_wq            6               root            txt             unknown                         /proc/6/exe (Permission denied)
mm_percpu_wq            6               root            NOFD                                            /proc/6/fd (Permission denied)
ksoftirqd/0             7               root            cwd             unknown                         /proc/7/cwd (Permission denied)
ksoftirqd/0             7               root            rtd             unknown                         /proc/7/root (Permission denied)
ksoftirqd/0             7               root            txt             unknown                         /proc/7/exe (Permission denied)
```

Run the command with root permission

```
$ sudo ./hw1| head -n 20
COMMAND         PID             USER            FD              TYPE            NODE            NAME
systemd         1               root            cwd             DIR             2               /
systemd         1               root            rtd             DIR             2               /
systemd         1               root            txt             REG             23734219                /lib/systemd/systemd
systemd         1               root            DEL             REG             23724387                /lib/x86_64-linux-gnu/libm-2.27.so
systemd         1               root            mem             REG             23724289                /lib/x86_64-linux-gnu/libudev.so.1
systemd         1               root            mem             REG             23724380                /lib/x86_64-linux-gnu/libgpg-erro
systemd         1               root            mem             REG             23724264                /lib/x86_64-linux-gnu/libjson-c.so
systemd         1               root            mem             REG             24775614                /usr/lib/x86_64-linux-gnu/libargo
systemd         1               root            mem             REG             23724307                /lib/x86_64-linux-gnu/libdevmappe
systemd         1               root            mem             REG             23725490                /lib/x86_64-linux-gnu/libattr.so.1
systemd         1               root            mem             REG             23725195                /lib/x86_64-linux-gnu/libcap-ng.so
systemd         1               root            mem             REG             23724192                /lib/x86_64-linux-gnu/libuuid.so.1
systemd         1               root            DEL             REG             23724382                /lib/x86_64-linux-gnu/libdl-2.27.s
systemd         1               root            mem             REG             23724639                /lib/x86_64-linux-gnu/libpcre.so.3
systemd         1               root            DEL             REG             23724459                /lib/x86_64-linux-gnu/libpthread-2
systemd         1               root            mem             REG             24772961                /usr/lib/x86_64-linux-gnu/liblz4.s
systemd         1               root            mem             REG             23725142                /lib/x86_64-linux-gnu/liblzma.so.5
systemd         1               root            mem             REG             23724150                /lib/x86_64-linux-gnu/libidn.so.11
systemd         1               root            mem             REG             24772626                /usr/lib/x86_64-linux-gnu/libip4t
```

© 2008—2022 Chun-Ying Huang, Security and Systems Laboratory, Department of Computer Science, National Chiao Tung University.