try / README1.md

Ashleylinn Update README1.md f5430e0 · now

187 lines (150 loc) · 8.79 KB

Preview Code Blame Raw

Recreating the System-Wide FD Tables

This program will display the tables used by the OS which generate the process table, system-wide FD table, Vnodes FD table, a composed view of the preveious table, and the table summarizing the number of FDs open per processes. It will be worked in the Linux OS (IA3170).

How I solve the problem

I solve this problem by breaking all the command line arguments to each source code so that I can solve them one by one. Since splitting all the function to a single source code makes me feel easier to know what I am doing. And I first start with the composite table (composite.c) as it has all the elements (PID, FD, Filename, and Inode). So, when I do process FD table, system-wide FD table, and Vnodes table, I just modify some parts of it, which makes it easier for me. And I take the argument of PID so that i know if there is specified PID or not. If yes, then it will open up that particular PID from /proc to get all the FDs. Otherwise, it will read all directories in /proc and then get PIDs and FDs. For the summary and threshold table, they are just having the different format with the table, so I modify a bit. The threshold should return the processes with FD if the fd is larger than the specified threshold. When i run the program, I acknowlege that if I dont keep track the first process, then it will not show the PID(FD) properly, so I add a variable to keep track that only add the comma after the process.

Implementation

`"functions.h"`

- create my own header file that stores all of my function declarations
- connect all of my source code together

`/proc`

- Get information for each processes and FDs
- Get the FDs for specified PID in /proc/PID/fd

`<dirent.h>`

- Used for the structure of dirent
- Get the name of entry

`<sys/types.h>`

- ssize_t is defined in here
- readlink() returns the number of bytes and if not success it will return -1 as error

`<sys/stat.h>`

- Used for the structure of stat
- Get the inode

Functions Documentation

`void composite_table(int PID)`

- This function will only display the composed FD table (Index, PID, FD, Filenmae, and Inodes)
- If there is a specified PID, then it will use that PID to retrieve informations
- If there is no specified PID, then it will retrieve information from all PIDs
- Using stat to get inode
- Also handle arguments of --output_TXT or --output_binary to write the composed table to compositionTable.txt/compositionTable.bin

`void perProcess_table(int PID)`

- This function will only display the process FD table (PID and FD)
- If there is a specified PID, then it will use that PID to retrieve informations

- If there is no specified PID, then it will retrieve information from all PIDs

`void sysWide_table(int PID)`

- This function will display only the system-wide FD table (PID, FD, and Filenmae)
- If there is a specified PID, then it will use that PID to retrieve informations
- If there is no specified PID, then it will retrieve information from all PIDs

`void Vnode_table(int PID)`

- This function display only the FD tables with Inodes
- If there is a specified PID, then it will use that PID to retrieve informations
- If there is no specified PID, then it will retrieve information from all PIDs
- Using stat to get inode

`void summary_table()`

- This function display the summarize table with PID (FD)
- It will display all the processes and the FDs open per process

`void threshold_flag(int threshold)`

- This function display the processs that are larger than the value X provided.
- threshold should be larger than 0

`int isPID(char *arg)`

- This function checks for if the user has specified a PID
- return 1 if user inpu a PID

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

- The main function handles all the command line arguments
- Calls other function to display a specific FD table

Pseudo-code

- Parse the command line arguments:
  - Check if there is a specified PID and is a numeric character
  - And check if there is other flags (--composite, --per-process, --systemWide, --Vnodes, --summary, --threshold, --output_TXT, --output_binary)
- If there is no command line arguments

    - show the default behavior of setting compos = 1
- If there is specified threshold flag
    - if is greater than 0 call threshold_flag(threshold)
- If there is specified PID
    - If is with other flags
        - only call the tables that being selected
    - If no other flags
        - display all the tables (composite, per-process, systemWide, and Vnodes)
- If no specified PID (PID = -1)
    - call the tables according to the flags
        - if perProcess = 1 then will call the perProcess_table(PID)
        - if sysWide = 1 then will call the systemWide_table(PID)
        - if Vnode = 1 then will call Vnode_table(PID)
        - if compos or TXT or binary equals to 1 then will call composite_table(PID, TXT, binary)
        - if summary = 1 then will call summary_table()

How to run the program

1. Compile with Makefile

    ```
    make
    ```

- I use -D_GNU_SOURCE in CC which close the warning for readlink()

2. Run

    ```
     ./showFDtables or
     ./showFDtables with PID / flags
    (ex, ./showFDtables 11873 --composite, ./showFDtables --per-process)
    ```

3. Clean

    ```
    make clean
    ```

- will remove all the object files, txt, and bin.

CLAs

- `--per-process`
    - Display only the process FD table
    - PID and FD
- `--systemWide`
    - Display only the system-wide FD table
    - PID, FD, and Filename
- `--Vnodes`
    - Display only the Vnodes FD table
    - FD and Inodes
- `--composite`
    - Display only the composed table
    - Index, PID, FD, Filenmae, and Inodes
    - Default if there is no specified argument (ex, ./showFDtables)
- `--summary`
    - Display the table that summarize the number of FDs open per process
- `--threshold=X`
    - X is an integer
    - Processes that have a number of FD larger than X will be flagged in the output
- `--output_TXT`
    - Save the composite table in text to the file "compositeTable.txt"
    - will also print out the composed table
- `--output_binary`
    - Save the composite table in binary format to "compositeTable.bin"
    - will also print out the composed table

Test cases

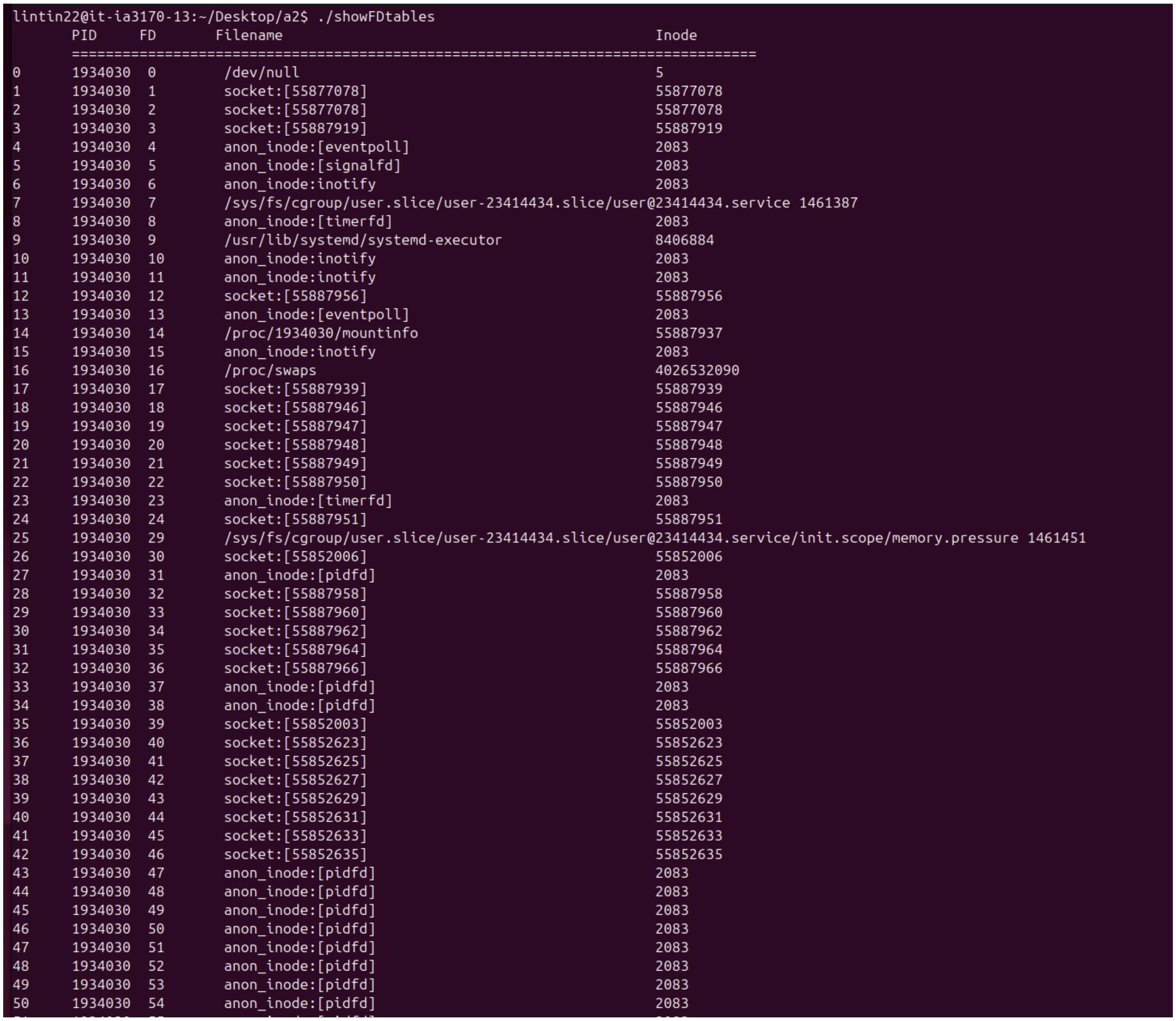
- `./showFDtables`
    - it shows the default behavior which is the composed table
    - if having typo for command line arguments, it will just do the default behavior which shows the composite table
- `./showFDtables 1134030`
    - it shows all the tables with the PID of 1134030

    - if PID not exists then it will not show any table which only the titles (PID, FD, Filenmae, Inode) and lines (=====) and will display the message of "Cannot open the directory: No such file or directory"
- `./showFDtables --per-process --composite`
    - it just shows the per process FD table and the composed table
- `./showFDtables --per-process --systemWide -- Vnodes --composite`
    - it will shows the tables with this following order
- `./showFDtables --composite --per-process`
    - it will show the perProcess table first then the composite table
    - All the table is displaying by this order (perProcess, systemWide, Vnodes, Composite)
- `./showFDtables --summary`
    - it shows the summarized table of all the processes and FDs with the format PID(FD)
- `./showFDtables --threshold=20`
    - it will shows all the process that have FDs larger than 20
- `./showFDtables --composite --threshold=15`
    - it shows offending processes that has FD larger than 15 then display the composed table
- `./showFDtables --threshold=100000000`
    - if there is a specifeid threshold that no FDs are larger then it will shows "No offending"
- `./showFDtables --output_TXT`
    - it will display the composed table
    - it will write the composed table to the compositeTable.txt
    - if it does not successfully open then it will send an error "Cannot open the file"
- `./showFDtables --output_binary`
    - it will display the composed table
    - it will write the composed table to the compostieTable.bin
    - if it does not successfully open then it will send an error "Cannot open the file"

Example

`./showFDtables`

(it will show the composed table since there is no argument)



```
lintin22@it-ia3170-13:~/Desktop/a2$ ./showFDtables
        PID      FD       Filename                                             Inode
        ========================================================================================
0       1934030  0        /dev/null                                            5
1       1934030  1        socket:[55877078]                                    55877078
2       1934030  2        socket:[55877078]                                    55877078
3       1934030  3        socket:[55887919]                                    55887919
4       1934030  4        anon_inode:[eventpoll]                               2083
5       1934030  5        anon_inode:[signalfd]                                2083
6       1934030  6        anon_inode:inotify                                   2083
7       1934030  7        /sys/fs/cgroup/user.slice/user-23414434.slice/user@23414434.service 1461387
8       1934030  8        anon_inode:[timerfd]                                 2083
9       1934030  9        /usr/lib/systemd/systemd-executor                    8406884
10      1934030  10       anon_inode:inotify                                   2083
11      1934030  11       anon_inode:inotify                                   2083
12      1934030  12       socket:[55887956]                                    55887956
13      1934030  13       anon_inode:[eventpoll]                               2083
14      1934030  14       /proc/1934030/mountinfo                              55887937
15      1934030  15       anon_inode:inotify                                   2083
16      1934030  16       /proc/swaps                                          4026532090
17      1934030  17       socket:[55887939]                                    55887939
18      1934030  18       socket:[55887946]                                    55887946
19      1934030  19       socket:[55887947]                                    55887947
20      1934030  20       socket:[55887948]                                    55887948
21      1934030  21       socket:[55887949]                                    55887949
22      1934030  22       socket:[55887950]                                    55887950
23      1934030  23       anon_inode:[timerfd]                                 2083
24      1934030  24       socket:[55887951]                                    55887951
25      1934030  29       /sys/fs/cgroup/user.slice/user-23414434.slice/user@23414434.service/init.scope/memory.pressure 1461451
26      1934030  30       socket:[55852006]                                    55852006
27      1934030  31       anon_inode:[pidfd]                                   2083
28      1934030  32       socket:[55887958]                                    55887958
29      1934030  33       socket:[55887960]                                    55887960
30      1934030  34       socket:[55887962]                                    55887962
31      1934030  35       socket:[55887964]                                    55887964
32      1934030  36       socket:[55887966]                                    55887966
33      1934030  37       anon_inode:[pidfd]                                   2083
34      1934030  38       anon_inode:[pidfd]                                   2083
35      1934030  39       socket:[55852003]                                    55852003
36      1934030  40       socket:[55852623]                                    55852623
37      1934030  41       socket:[55852625]                                    55852625
38      1934030  42       socket:[55852627]                                    55852627
39      1934030  43       socket:[55852629]                                    55852629
40      1934030  44       socket:[55852631]                                    55852631
41      1934030  45       socket:[55852633]                                    55852633
42      1934030  46       socket:[55852635]                                    55852635
43      1934030  47       anon_inode:[pidfd]                                   2083
44      1934030  48       anon_inode:[pidfd]                                   2083
45      1934030  49       anon_inode:[pidfd]                                   2083
46      1934030  50       anon_inode:[pidfd]                                   2083
47      1934030  51       anon_inode:[pidfd]                                   2083
48      1934030  52       anon_inode:[pidfd]                                   2083
49      1934030  53       anon_inode:[pidfd]                                   2083
50      1934030  54       anon_inode:[pidfd]                                   2083
```

References

- https://man7.org/linux/man-pages/man2/readlink.2.html
- https://man7.org/linux/man-pages/man2/stat.2.html
- https://www.tutorialspoint.com/how-to-align-the-output-using-justificationsin-c-language
- https://man7.org/linux/man-pages/man5/proc.5.html
- https://stackoverflow.com/questions/9449241/where-is-path-max-defined-in-linux