Universitatea Tehnică a Moldovei

Facultatea Calculatoare Informatică şi Microelectronică

Departamentul Ingineria Software și Automatică

**RAPORT**

Lucrarea de laborator nr. 5

# La disciplina „ Sistemul de operare”

Tema: "Monitorizarea proceselor"

A efectuat: st. gr. SI-211 A. Chihai

A verificat: V. Beșliu

Chișinău – 2023

Lucrarea de laborator nr. 5

#### Scopul lucrarii: modurile de obținere de informații despre rularea proceselor, despre resursele utilizate de procese, prezentarea rezultatelor în diverse forme.

**Sarcină la laborator 5**

Executați următorii pași:

a) Toate script-urile și fișierele pentru extragerea rezultatelor le veți crea în subdirectorul lab5.

b) Scrieți script-uri, care vor rezolva următoarele probleme:

1) Aflaţi numărul de procese inițiaţializate de utilizatorul user și introduceți într-un fișier perechea „PID: comanda” a proceselor inițializate. Afișați conținutul acestui fișier şi numărul de procese.

2) Afișați PID-ul procesului, ultimul lansat (cu timpul de lansare).

3) Introduceți într-un fișier o listă cu PID-urile proceselor, pornite cu comenzile localizate în /sbin/. Afișați conținutul acest fișier.

4) Pentru fiecare proces, calculați diferența dintre memoria totală (statm: size) şi rezidentă (statm: resident) ale memoriei de proces (în pagini). Introduceți într-un fișier linii de tip „PID:diferența”, sortate în ordinea descrescătoare a acestor diferențe. Afișați conținutul acestui fișier.

5) Pentru toate procesele înregistrate, la moment, în sistem, introduceți într-un fişier linii de tip ProcessID=PID:Parent\_ProcessID=PPID:

Average\_Time=avg\_atom

Preluați valorile PPID și PID din fișierele status,

valoarea avg\_atom

(avg\_atom=se.sum\_exec\_runtime/nr\_switches)

din fișierele sched care se află în subdirectoarele, cu nume ce corespund PID proceselor în directorul /proc. Sortați aceste linii conform identificatorilor proceselor părinte. Rezultatul sortării introduceți într-un fișier și afișați-l.

6) În fișierul obținut în 5, după fiecare grup de înregistrări cu același identificator al procesului părinte, introduceți o linie de tip

Sum\_switches\_of\_ParentID=N is M, unde

N=PPID, iar M este suma calculată

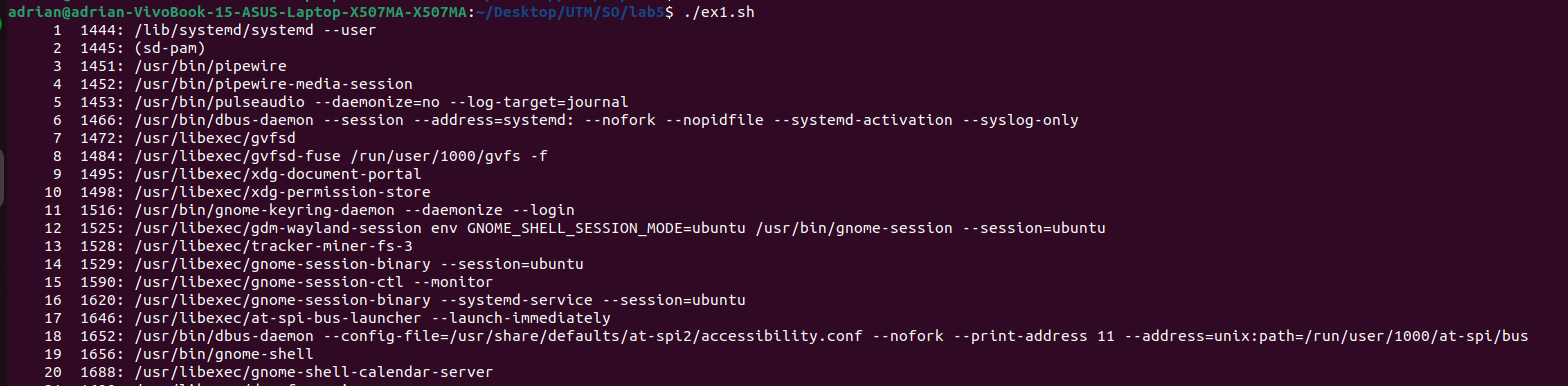
voluntary\_ctxt\_switches+

nonvoluntary\_ctxt\_switches din statuspentru acest proces. Afișați conținutul acestui fișier.

Mersul lucrări:

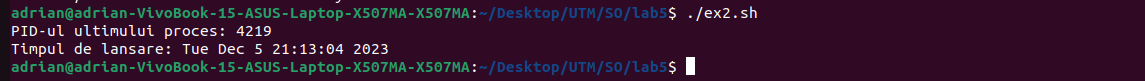
Aflaţi numărul de procese inițiaţializate de utilizatorul user și introduceți într-un fișier perechea „PID: comanda” a proceselor inițializate. Afișați conținutul acestui fișier şi numărul de procese.

#!/bin/bash  
output\_ps=$(ps -u $USER -o pid,cmd --no-headers)  
echo "$output\_ps" | awk '{print $1 ": " substr($0, index($0,$2))}' > procese.txt  
cat -n procese.txt



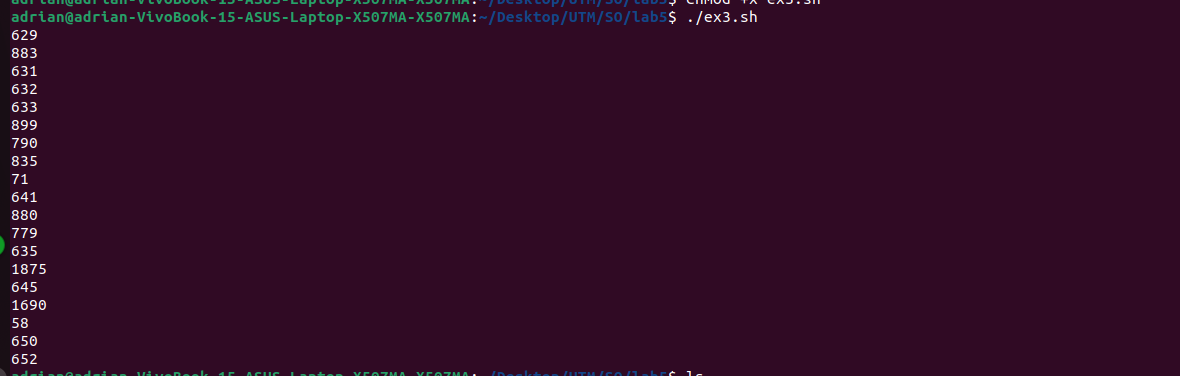
Afișați PID-ul procesului, ultimul lansat (cu timpul de lansare).

#!/bin/bash  
  
# Obține informații despre ultimul proces lansat  
ultimul\_proces=$(ps -eo pid,lstart --sort=start\_time | tail -n 1)  
  
# Extrage PID-ul și timpul de lansare utilizând awk  
pid=$(echo $ultimul\_proces | awk '{print $1}')  
timp\_lansare=$(echo $ultimul\_proces | awk '{$1=""; print $0}' | sed 's/^[ \t]\*//')  
  
# Afișează rezultatele  
echo "PID-ul ultimului proces: $pid"  
echo "Timpul de lansare: $timp\_lansare"



Introduceți într-un fișier o listă cu PID-urile proceselor, pornite cu comenzile localizate în /sbin/. Afișați conținutul acest fișier.

#!/bin/bash  
  
# Directorul în care sunt localizate comenzile  
director\_comenzi="/sbin/"  
  
# Numele fișierului în care vor fi stocate PID-urile  
fisier\_pid="pid\_file.txt"  
  
# Caută procesele din /sbin/ și obține PID-urile lor  
piduri=$(  
 for cmd in $director\_comenzi\*; do  
 pid=$(pgrep -o "$(basename $cmd)")  
 if [ -n "$pid" ]; then  
 echo "$pid"  
 fi  
 done  
)  
  
# Salvează PID-urile în fișierul specificat  
echo "$piduri" > "$fisier\_pid"  
  
# Afișează conținutul fișierului  
cat "$fisier\_pid"



Pentru fiecare proces, calculați diferența dintre memoria totală (statm: size) şi rezidentă (statm: resident) ale memoriei de proces (în pagini). Introduceți într-un fișier linii de tip „PID:diferența”, sortate în ordinea descrescătoare a acestor diferențe. Afișați conținutul acestui fișier.

#!/bin/bash  
  
# Numele fișierului în care vor fi stocate rezultatele  
fisier\_rezultate="diferente\_memorie.txt"  
  
# Obține PID-urile proceselor din /sbin/  
piduri=$(  
 for cmd in /sbin/\*; do  
 pid=$(pgrep -o "$(basename $cmd)")  
 if [ -n "$pid" ]; then  
 echo "$pid"  
 fi  
 done  
)  
  
# Parcurge fiecare PID și calculează diferența între memoria totală și memoria rezidentă  
for pid in $piduri; do  
 # Obține informații despre memoria procesului  
 size=$(awk '/VmSize/{print $2}' "/proc/$pid/status")  
 resident=$(awk '/VmRSS/{print $2}' "/proc/$pid/status")  
  
 # Calculează diferența  
 diferenta=$((size - resident))  
  
 # Introduce în fișier linia "PID:diferenta"  
 echo "$pid:$diferenta" >> "$fisier\_rezultate"  
done  
  
# Sortează fișierul în ordinea descrescătoare a diferențelor  
sort -t':' -k2,2nr -o "$fisier\_rezultate" "$fisier\_rezultate"  
  
# Afișează conținutul fișierului  
cat "$fisier\_rezultate"



Pentru toate procesele înregistrate, la moment, în sistem, introduceți într-un fişier linii de tip ProcessID=PID:Parent\_ProcessID=PPID:

Average\_Time=avg\_atom

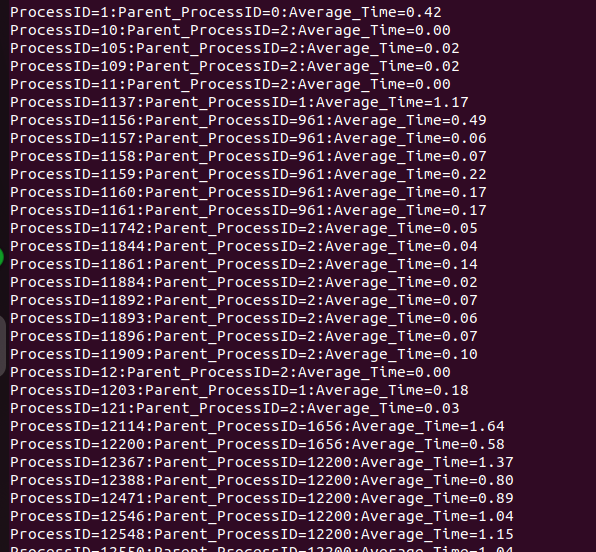
Preluați valorile PPID și PID din fișierele status,

valoarea avg\_atom

(avg\_atom=se.sum\_exec\_runtime/nr\_switches)

din fișierele sched care se află în subdirectoarele, cu nume ce corespund PID proceselor în directorul /proc. Sortați aceste linii conform identificatorilor proceselor părinte. Rezultatul sortării introduceți într-un fișier și afișați-l.

#!/bin/fish  
  
# Directorul în care se află fișierele de informații despre procese (/proc)  
set proc\_dir "/proc"  
  
# Fișierul în care vom scrie rezultatele  
set output\_file "rezultate.txt"  
  
# Cautăm toate subdirectoarele în /proc care reprezintă procese  
for pid\_dir in $proc\_dir/[0-9]\*  
 set pid (basename "$pid\_dir")  
  
 # Verificăm dacă fișierul status există pentru procesul curent  
 set status\_file "$pid\_dir/status"  
 if test -f "$status\_file"  
 # Extragem informațiile necesare din fișierul status  
 set ppid (grep 'PPid' "$status\_file" | awk '{print $2}')  
 set sum\_exec\_runtime (grep 'se.sum\_exec\_runtime' "$status\_file" | awk '{print $2}')  
 set nr\_switches (grep 'nr\_switches' "$status\_file" | awk '{print $2}')  
  
 # Calculăm avg\_atom  
 if test "$nr\_switches" -ne 0  
 set avg\_atom (math "$sum\_exec\_runtime" / "$nr\_switches")  
 else  
 set avg\_atom 0  
 end  
  
 # Adăugăm informațiile într-un fișier  
 echo "ProcessID=$pid:Parent\_ProcessID=$ppid:Average\_Time=$avg\_atom" >> "$output\_file"  
 end  
end  
  
# Sortăm rezultatele după identificatorii proceselor părinte (PPID)  
sort -t "=" -k4 -n -o "$output\_file"  
  
# Afișăm rezultatele  
echo "Rezultatele au fost salvate în $output\_file"  
cat "$output\_file"



În fișierul obținut în 5, după fiecare grup de înregistrări cu același identificator al procesului părinte, introduceți o linie de tip

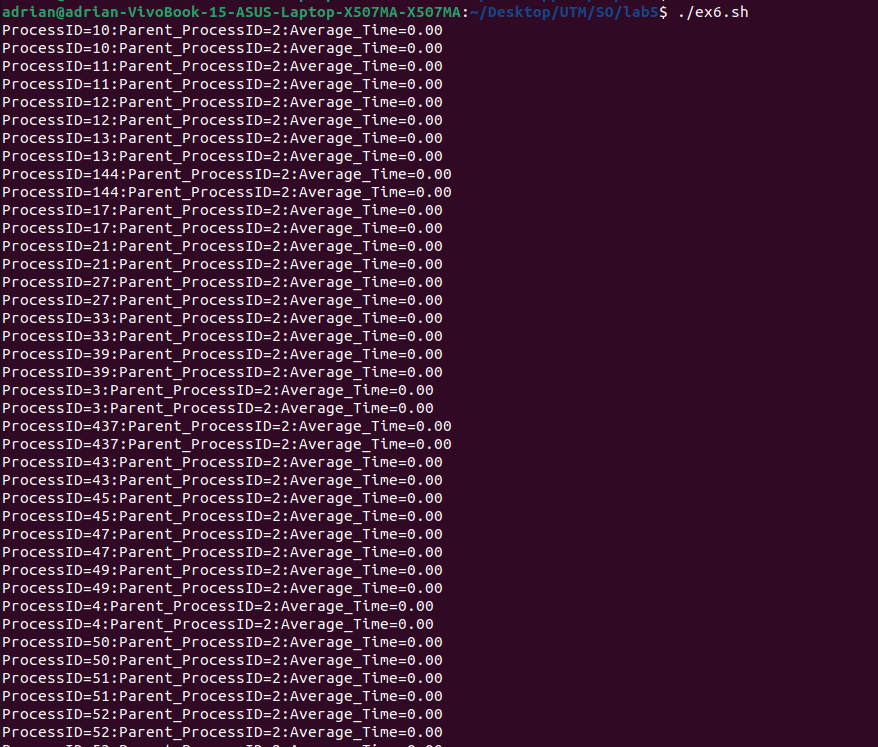
Sum\_switches\_of\_ParentID=N is M, unde

N=PPID, iar M este suma calculată

voluntary\_ctxt\_switches+

nonvoluntary\_ctxt\_switches din statuspentru acest proces. Afișați conținutul acestui fișier.

#!/bin/bash  
  
input\_file="rezultate.txt"  
output\_file="output\_file\_modified.txt"  
  
# Sortarea fișierului după PPID  
sort -t= -k4,4n -k8,8n "$input\_file" > "$output\_file"  
  
# Iterarea prin fișier și adăugarea liniei Sum\_switches\_of\_ParentID  
current\_parent\_id=""  
sum\_switches=0  
  
while IFS= read -r line  
do  
 ppid=$(grep -oP 'Parent\_ProcessID=\K\d+' <<< "$line")  
 switches=$(grep -oP 'voluntary\_ctxt\_switches:\s+\K\d+' <<< "$line")  
  
 if [ "$ppid" = "$current\_parent\_id" ]; then  
 sum\_switches=$((sum\_switches + switches))  
 else  
 echo "Sum\_switches\_of\_ParentID=$current\_parent\_id is $sum\_switches" >> "$output\_file"  
 current\_parent\_id="$ppid"  
 sum\_switches="$switches"  
 fi  
done < "$output\_file"  
  
# Adăugarea ultimei linii pentru ultimul grup  
echo "Sum\_switches\_of\_ParentID=$current\_parent\_id is $sum\_switches" >> "$output\_file"  
  
# Afișarea conținutului fișierului modificat  
cat "$output\_file"



Concluzie: În rezumat, gestionarea și monitorizarea proceselor într-un sistem informatic reprezintă o componentă esențială pentru întelegerea și optimizarea utilizării resurselor. Prin implementarea unor sarcini precum identificarea și înregistrarea proceselor, analiza variațiilor în consumul de memorie și evaluarea performanțelor în funcție de timpul de execuție, obținem o perspectivă detaliată asupra activității sistemului. Această monitorizare nu doar furnizează informații cruciale pentru identificarea problemelor de performanță, ci contribuie și la optimizarea generală a resurselor și proceselor. Prin acest demers, se realizează un mediu informatic eficient și stabil, asigurându-ne că sistemul operează la capacitatea sa maximă, cu costuri minime și o fiabilitate sporită.