

1. Задание

Основная цель лабораторной работы - знакомство с системными инструментами анализа производительности и поведения программ. Для этого предлагается для выданной по варианту программы выяснить следующую информацию:

1. Количество потоков создаваемое программой;
2. Список файлов и сетевых соединений с которыми работает программа
3. Карту памяти процесса;
4. Содержимое передаваемых по сети данных;
5. Построить графики:

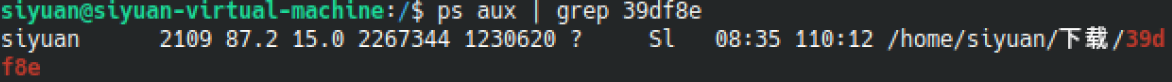
* Потребления программой cpu;
* Нагрузки генерируемой программой на подсистему ввода-вывода;
* Нагрузки генерируемой программой на сетевую подсистему.
* Смены состояния исполнения потоков;

1. Коды
   1. Количество потоков создаваемое программой;

At the first we find the <pid> according to the file name

(1)ps aux | grep 39df8e

(2)ps -ef | grep 39df8e



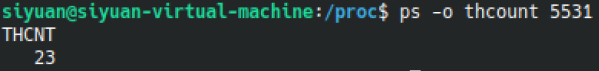
*Find the info from /proc according to pid.*

cat /proc/<pid>/status | grep Threads



Or we can also use command “ps”:

ps -o thcount <pid>



* 1. Список файлов и сетевых соединений с которыми работает программа

Список файлов:

strace <program\_name> 2>&1 | grep 'openat'

文本

描述已自动生成

Список сетевых соединений

lsof -ai -p <pid>

文本

描述已自动生成

* 1. Карту памяти процесса;

pmap <pid>

文本

描述已自动生成

屏幕上有字

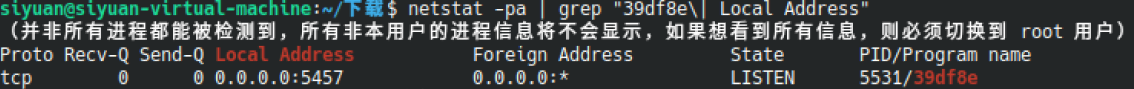
描述已自动生成

If we want more and clear info, we need -x.

* 1. Содержимое передаваемых по сети данных;

At the first we need to find the ports which our program uses.

netstat -pa | grep “<filename> \| Local Address”



Let's look at the transmitted packets using the found ports using the tcpdump program.

Sudo tcpdump -I any -A port 5457

文本

描述已自动生成

* 1. Построить графики:
* Потребления программой cpu;

#!/bin/bash

# process id to monitor

pid=$1

if [ -z $1 ]; then

  echo "ERROR: Process ID not specified."

  echo

  echo "Usage: $(basename "$0") <PID>"

  exit 1

fi

# check if process exists

kill -0 $pid > /dev/null 2>&1

pid\_exist=$?

if [ $pid\_exist != 0 ]; then

  echo "ERROR: Process ID $pid not found."

  exit 1

fi

current\_time=$(date +"%Y\_%m\_%d\_%H%M")

dir\_name="data/${pid}-${current\_time}"

csv\_filename="${dir\_name}/metrics.csv"

# create data directory

mkdir -p $dir\_name

# Read collected metrices from the CSV file and plot graphs

#

# This function will end script execution.

#

# This function is to be called after an interrupt like SIGINT or SIGKILL

# is received.

#

function plotGraph() {

  # bring cursor to next line after interrupt

  echo

  # plot graphs if there is a data file

  if [ -f $csv\_filename ]; then

echo "Plotting graphs..."

gnuplot <<- EOF

   # Output to png with a font size of 10, using pngcairo for anti-aliasing

   set term pngcairo size 1024,800 noenhanced font "Helvetica,10"

   # Set border color around the graph

   set border ls 50 lt rgb "#939393"

   # Hide left and right vertical borders

   set border 16 lw 0

   set border 64 lw 0

   # Set tic color

   set tics nomirror textcolor rgb "#939393"

   # Set horizontal lines on the ytics

   set grid ytics lt 1 lc rgb "#d8d8d8" lw 2

   # Rotate x axis lables

   set xtics rotate

   # Set graph size relative to the canvas

   set size 1,0.85

   # Set separator to comma

   set datafile separator ","

   # Move legend to the bottom

   set key bmargin center box lt rgb "#d8d8d8" horizontal

   # Plot graph,

   # xticlabels(1) - first column as x tic labels

   # "with lines" - line graph

   # "smooth unique"

   # "lw 2" - line width

   # "lt rgb " - line style color

   # "t " - legend labels

   #

   # CPU and memory usage

   set output "${dir\_name}/cpu-mem-usage.png"

   set title "CPU and Memory Usage for Proces ID $pid"

   plot "$csv\_filename" using 2:xticlabels(1) with lines smooth unique lw 2 lt rgb "#4848d6" t "CPU Usage %",\

    "$csv\_filename" using 3:xticlabels(1) with lines smooth unique lw 2 lt rgb "#b40000" t "Memory Usage %"

EOF

  fi

  echo "Done!"

  exit 0

}

# add SIGINT & SIGTERM trap

trap "plotGraph" SIGINT SIGTERM SIGKILL

echo "Writing data to CSV file $csv\_filename..."

touch $csv\_filename

# write CSV headers

echo "Time,CPU,Memory,TCP Connections,Thread Count" >> $csv\_filename

# check if process exists

kill -0 $pid > /dev/null 2>&1

pid\_exist=$?

# collect until process exits

while [ $pid\_exist == 0 ]; do

  # check if process exists

  kill -0 $pid > /dev/null 2>&1

  pid\_exist=$?

  if [ $pid\_exist == 0 ]; then

# read cpu and mem percentages

timestamp=$(date +"%b %d %H:%M:%S")

cpu\_mem\_usage=$(top -b -n 1 | grep -w -E "^ \*$pid" | awk '{print $9 "," $10}')

# write CSV row

echo "$timestamp,$cpu\_mem\_usage" >> $csv\_filename

sleep 1

  fi

done

# draw graph

plotGraph

Then we run it:

./process-printer.sh <PID>

We will get result like this:

图表, 折线图

描述已自动生成

Нагрузки на подсистему ввода-вывода

pidstat -d -p <pid> 1

图形用户界面

描述已自动生成

图表, 折线图

描述已自动生成