



Nome: _____ RA: _____

- 1) Liste o PID dos 5 (cinco) processos mais recentes instanciados no seu sistema operacional.

\$ ps a

- 2) Utilize um processo para listar a quantidade de memória do seu sistema operacional.

\$ cat /proc/meminfo

- 3) Inicie o editor de texto 'gedit' a partir do terminal. Finalize o processo com o comando 'kill'.

\$ ps aux | grep nano

\$ kill 11765

```
sta 1/Lab2$ ps aux | grep nano
a2408740 11765 0.0 0.0 6504 2
a2408740 11833 0.0 0.0 6504 2
a2408740 13080 0.0 0.0 6504 2
a2408740 13104 0.0 0.0 6504 2
a2408740 13152 0.0 0.0 6504 2
a2408740 22832 0.0 0.0 6568 2
a2408740@pxe-ubuntu:/media/a2408740/
sta 1/Lab2$ ^C
a2408740@pxe-ubuntu:/media/a2408740/
sta 1/Lab2$ kill 11765
```

- 4) Qual o limite de processos que podem ser instanciados pelo seu SO? (Dica: utilize o comando 'cat /proc/sys/kernel/pid_max')

```
sta 1/Lab2$ cat /proc/sys/kernel/pid_max
4194304
```

- 5) Uma **syscall** é uma função do SO que o programa pode invocar. Observe a chamada da syscall a seguir:

package syscall;

```

import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;

public class Syscall {

    public static void main(String [] args) {

        try {
            Runtime r = Runtime.getRuntime();
            Process p = r.exec("uname -a");
            p.waitFor();
            BufferedReader b = new BufferedReader(new
            InputStreamReader(p.getInputStream()));
            String line = "";

            while ((line = b.readLine()) != null) {
                System.out.println(line);
            }

            b.close();
        } catch (IOException e){
            System.out.println(e.getMessage());
        } catch (InterruptedException e){
            System.out.println(e.getMessage());
        }
    }
}

```

- a) Qual é a syscall? uname
- b) Qual é o número da syscall?
- c) Qual o propósito dessa syscall? Mostrar informações do sistema
- d) Modifique o programa para acionar a syscall 'sysinfo'

6) O comando **kill** é uma **syscall** usada para enviar **sinais** para processos no Linux. (\$man 7 signal) (Nota: syscalls geralmente não são chamadas diretamente, mas sim com funções encapsuladoras da glibc, ou outra biblioteca (man 2 syscalls). Portanto, existe uma função kill (glibc) que invoca a syscall kill (kernel). Por simplicidade, iremos considerar ambos como uma **syscall**).

kill -l: é uma **syscall** que lista os sinais disponíveis no seu sistema operacional Linux. Informe os 5 (cinco) primeiros sinais listados.

\$ 1) SIGHUP 2) SIGINT 3) SIGQUIT 4) SIGILL 5) SIGTRAP

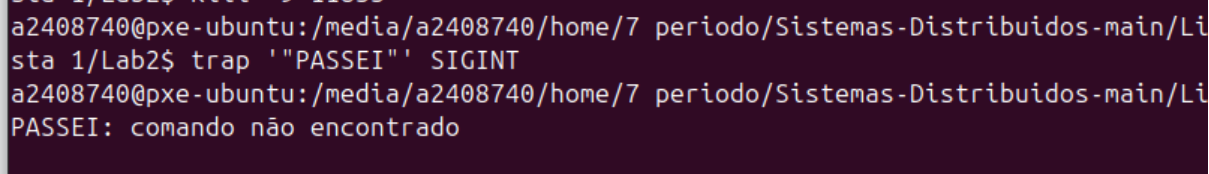
7) Um sinal pode ser disparado com um simples **CTRL+C**. Esta sequência envia um **SIGINT** para o processo e ele finaliza.

Abra um editor de texto de sua escolha e finalize o processo com os sinais a seguir:

```
$ kill -<SIGNAL> <PID>
$
$ #SIGINT: Interrupcao do teclado
$ kill -2 123
$ kill -SIGINT 123
$
$ #SIGTERM: Solicita termino normal do processo
$ kill -15 123
$ kill -SIGTERM 123
$
$ #SIGKILL: Solicita o termino imediato do processo
$ kill -9 123
$ kill -SIGKILL 123
```

8) O comando **trap** captura **sinais**. Exemplo:

```
$ trap "“Passei por aqui”" SIGINT
$ #Ctrl+C = SIGINT
$ ^CPassei por aqui
$
$ #Listar as traps
$ trap
$
$ #Remover a acao da trap
$ trap '' SIGINT
$
$ #Excluir a trap
$ trap SIGINT
```



```
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
sta 1/Lab2$ trap '"PASSEI"' SIGINT
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
PASSEI: comando não encontrado
```

Utilize o script Bash a seguir para exibir uma mensagem ao final da execução do script.

```
#!/bin/bash
```

```
function minha_trap {
    echo "Passei por aqui"; exit
}
trap minha_funcao SIGINT

while true
do
    echo .
    sleep 1
done
```

```
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
sta 1/Lab2$ ls
aula2.sh      Cliente.java  JASONObject.java  Servidor.class
Cliente.class  fortune-br.txt 'Servidor$FileReader.class' Servidor.java
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
sta 1/Lab2$ chmod u+x aula2.sh
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
sta 1/Lab2$ ls
aula2.sh      Cliente.java  JASONObject.java  Servidor.class
Cliente.class  fortune-br.txt 'Servidor$FileReader.class' Servidor.java
a2408740@pxe-ubuntu:/media/a2408740/home/7 periodo/Sistemas-Distribuidos-main/Li
sta 1/Lab2$ ./aula2.sh
.
.
.
.
.
.
.
.
.
.
^C./aula2.sh: linha 1: minha_funcao: comando não encontrado
.
```

```
groups: cannot find name for group ID 1024
a2408740@pxe-ubuntu:/media/a2408740/home/7
sta 1/Lab2$ ps aux | grep aula2
a2408740  33476  0.0  0.0  7344  3360 pt
aula2.sh
a2408740  33918  0.0  0.0  6568  2240 pt
a2408740@pxe-ubuntu:/media/a2408740/home/7
sta 1/Lab2$ kill -9 33476
a2408740@pxe-ubuntu:/media/a2408740/home/7
sta 1/Lab2$
```

Para encerrar ->

9) Utilize pipes no Linux para exibir o PID do processo mais recente em execução.

processoA | processoB | processoC

\$ gedit &

\$ ps aux | grep gedit

10) Pesquise no arquivo **/etc/services** qual a porta associada aos seguintes serviços:

SMTP:

FTP: 21

SSH:22

TELNET:23

WHOIS: 43

HTTP: 80

BGP: 179