

1. Войдите под пользователем user1 из практики 2 (su - user1)

2. Подсчитайте количество процессов, имеющих несколько потоков выполнения

```
ps -eLo pid | sort | uniq -c | awk '$1 > 1 {print}' | wc -l
```

10

3. Запустите top и настройте вывод полей с информацией о процессе следующим образом:

```
user2@eltex-practice2-pg1-v8: ~ 80x24
top - 02:36:20 up 4 days, 15 min, 1 user, load average: 0.00, 0.00, 0.00
Tasks: 107 total, 1 running, 106 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.3 us, 0.3 sy, 0.0 ni, 99.5 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3916.0 total, 1964.7 free, 499.6 used, 1747.8 buff/cache
MiB Swap: 3185.0 total, 3184.7 free, 0.3 used. 3416.4 avail Mem
```

PID	USER	RUSER	PR	NI	S	%CPU	%MEM	TIME+	COMMAND
1	root	root	20	0	S	0.0	0.3	4:56.96	systemd
2	root	root	20	0	S	0.0	0.0	0:00.03	kthreadd
3	root	root	20	0	S	0.0	0.0	0:00.00	pool_workqueue_relea+
4	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/R-rcu_g
5	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/R-rcu_p
6	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/R-slab_
7	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/R-netns
10	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/0:0H-events_+
12	root	root	0	-20	I	0.0	0.0	0:00.00	kworker/R-mm_pe
13	root	root	20	0	I	0.0	0.0	0:00.00	rcu_tasks_kthread
14	root	root	20	0	I	0.0	0.0	0:00.00	rcu_tasks_rude_kthre+
15	root	root	20	0	I	0.0	0.0	0:00.00	rcu_tasks_trace_kthr+
16	root	root	20	0	S	0.0	0.0	0:00.11	ksoftirqd/0
17	root	root	20	0	I	0.0	0.0	0:02.57	rcu_preempt
18	root	root	rt	0	S	0.0	0.0	0:01.50	migration/0
19	root	root	-51	0	S	0.0	0.0	0:00.00	idle_inject/0
20	root	root	20	0	S	0.0	0.0	0:00.00	cpuhp/0

4. В другом терминальном окне выполните команду passwd и оставьте ее в состоянии запроса текущего пароля

```
user2@eltex-practice2-pg1-v8: ~$ passwd
Changing password for user2.
Current password: 
```

5. Перейдите в терминальное окно с top и выполните следующие действия:

- выведите все процессы, для которых реальным пользователем является пользователь, которым вы вошли в сеанс;
- найдите процесс, запущенный командой passwd;
- отправьте этому процессу сигналы 15 (SIGTERM), 2 (SIGINT), 3 (SIGQUIT), 9 (SIGKILL)

PID	USER	PR	NI	VIRT	RES	SHR	S	%CPU	%MEM	TIME+	COMMAND
46812	user2	20	0	8776	5504	3840	S	0.0	0.1	0:00.02	bash
46983	user2	20	0	8644	5504	3840	S	0.0	0.1	0:00.01	bash
<b>47024</b>	<b>user2</b>	<b>20</b>	<b>0</b>	<b>11920</b>	<b>5760</b>	<b>3712</b>	<b>R</b>	<b>0.0</b>	<b>0.1</b>	<b>0:00.00</b>	<b>top</b>

```

user2@eltex-practice2-pg1-v8: ~
top - 02:56:17 up 4 days, 35 min, 2 users, load average: 0.02, 0.01, 0.00
Tasks: 112 total, 1 running, 111 sleeping, 0 stopped, 0 zombie
%Cpu(s): 0.1 us, 0.0 sy, 0.0 ni, 99.9 id, 0.0 wa, 0.0 hi, 0.0 si, 0.0 st
MiB Mem : 3916.0 total, 1952.8 free, 511.1 used, 1748.1 buff/cache
MiB Swap: 3185.0 total, 3184.7 free, 0.3 used. 3404.9 avail Mem

  PID USER      PR  NI   VIRT   RES   SHR  S  %CPU  %MEM     TIME+ COMMAND
  47022 root        20   0    9172   3712  3456  S   0.0   0.1   0:00.00 passwd
  47023 user2       20   0   11920   5760  3712  R   0.0   0.1   0:00.00 top

user2@eltex-practice2-pg1-v8: ~ 94x16
passwd: Authentication token manipulation error
passwd: password unchanged
user2@eltex-practice2-pg1-v8:~$ passwd
Changing password for user2.
Current password:
New password:
Retype new password:
The password has not been changed.
New password:
Retype new password:
Sorry, passwords do not match.
passwd: Authentication token manipulation error
passwd: password unchanged
user2@eltex-practice2-pg1-v8:~$ passwd
Changing password for user2.
Current password: 

```

kill -15 PID  
kill -2 PID  
kill -3 PID  
kill -9 PID

passwd убил только SIGKILL

6. Выполните команду vim ~/file\_task3.txt и нажмите Ctrl-Z

```

user2@eltex-practice2-pg1-v8:~$ vim ~/file_task3.txt

[1]+  Stopped                  vim ~/file_task3.txt
user2@eltex-practice2-pg1-v8:~$ 

```

7. Выполните команду `sleep 600`, нажмите `Ctrl-Z` и выполните команду `jobs`

```
user2@eltex-practice2-pg1-v8:~$ sleep 600
^Z
[2]+  Stopped                  sleep 600
user2@eltex-practice2-pg1-v8:~$ jobs
[1]-  Stopped                  vim ~/file_task3.txt
[2]+  Stopped                  sleep 600
user2@eltex-practice2-pg1-v8:~$
```

8. Последнее задание (`sleep 600`) сделайте фоновым

`bg %2`

9. Измените число NICE у задания (`sleep 600`), сделав его равным 10

10 Проверьте, что число NICE у этого задания изменилось

```
user2@eltex-practice2-pg1-v8:~$ pgrep -a sleep
47070 sleep 600
user2@eltex-practice2-pg1-v8:~$ renice -n 10 47070
47070 (process ID) old priority 0, new priority 10
user2@eltex-practice2-pg1-v8:~$
```

```
[2] 47093
user2@eltex-practice2-pg1-v8:~$ renice -n 10 47093
47093 (process ID) old priority 0, new priority 10
user2@eltex-practice2-pg1-v8:~$ ps axl | grep sleep
0 1002  47093  47048 30 10 5684 2048 hrtim SN pts/0 0:00 sleep
0 1002  47096  47048 20 0 6544 2304 pipe_r S+ pts/0 0:00 grep
ep
user2@eltex-practice2-pg1-v8:~$
```

11 Сделайте задание `vim ~/file_task3.txt` активным и выйдите из редактора

`fg %1`

12 Отправьте сигнал 15 (SIGTERM) заданию `sleep 600` и выполните команду `jobs`

```
[1] 47105
user2@eltex-practice2-pg1-v8:~$ kill -15 47105
user2@eltex-practice2-pg1-v8:~$ jobs
[3]+  Stopped                  vim ~/file_task3.txt
[4]-  Terminated              sleep 600
user2@eltex-practice2-pg1-v8:~$ jobs
[3]+  Stopped                  vim ~/file_task3.txt
user2@eltex-practice2-pg1-v8:~$
```

13 Создайте перехватчик сигналов SIGINT и SIGQUIT внутри командного интерпретатора, который выводит сообщение «Меня голыми руками не возьмёшь!» (используйте встроенную команду trap) и отправьте сигналы самому себе

```
user2@eltex-practice2-pg1-v8:~$ trap 'echo "Меня голыми руками не возьмёшь!"' SIGINT SIGQUIT
Меня голыми руками не возьмёшь! ^C

Меня голыми руками не возьмёшь! ^\

user2@eltex-practice2-pg1-v8:~$
```

## Раздел 2

1. Создайте скрипт на языке bash с именем template\_task.sh,

```
#!/bin/bash

script_name=$(basename "$0")

if [ "$script_name" = "template_task.sh" ]; then
    echo "я бригадир, сам не работаю"
    exit 1
fi

log_file="report_${script_name%.*}.log" # Удаляем расширение .sh если есть

echo " $(date '+%Y-%m-%d %H:%M:%S') Скрипт запущен" >> "$log_file"

random_seconds=$(( RANDOM % 1771 + 30 )) # 1800-30+1=1771

sleep $random_seconds

minutes=$(( random_seconds / 60 ))

echo " $(date '+%Y-%m-%d %H:%M:%S') Скрипт завершился, работал $minutes минут" >> "$log_file"

exit 0
```

2. Создайте скрипт на языке bash с именем observer.sh

```
Observer.sh

#!/bin/bash

CONFIG_FILE="observer.conf"
LOG_FILE="observer.log"

if [ ! -f "$CONFIG_FILE" ]; then
    echo "$(date '+%Y-%m-%d %H:%M:%S') [ERROR] Конфигурационный файл $CONFIG_FILE не найден" >
    exit 1
fi

is_running() {
    script_name=$1
    for pid in /proc/[0-9]*; do
        if [ -f "$pid/cmdline" ]; then
            if grep -q "$script_name" "$pid/cmdline" 2>/dev/null; then
                return 0
            fi
        fi
    done
}
```

```

    fi
done
return 1
}

while IFS= read -r script || [ -n "$script" ]; do
    script_name=$(basename "$script")
    if is_running "$script_name"; then
        echo "$(date '+%Y-%m-%d %H:%M:%S') [INFO] Процесс $script_name уже запущен" >> "$LOG_"
    else
        nohup "$script" >/dev/null 2>&1 &
        echo "$(date '+%Y-%m-%d %H:%M:%S') [ACTION] Запущен процесс $script_name (PID: $!)" >>
    fi
done < "$CONFIG_FILE"

exit 0

```

3. Настройте запуск observer.sh посредством cron по расписанию – 1 раз в минуту

```
crontab -e
```

```
* * * * * /home/user2/observer.sh
```

4. Создайте несколько символических ссылок на файл template\_task.sh с различными именами (рабочие задачи), добавьте в файл конфигурации observer.conf соответствующие записи об этих задачах, включая исходный файл template\_task.sh

```
ln -s template_task.sh script1.sh
```

```
...
```

5. Соберите статистику работы в виде набора файлов report\_\*.log, observer.log, приложите их вместе с исходными текстами скриптов в качестве отчета в виде сжатого архива tar. Не забудьте остановить процесс, удалив задачу в cron!

```
mkdir observer_report
```

```
cp /path/to/scripts/observer.* .
```

```
...
```

```
tar -czf observer_report.tar.gz *
```