

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение  
высшего образования

**«КУБАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «КубГУ»)**

**Факультет компьютерных технологий и прикладной математики  
Кафедра информационных технологий**

**ОТЧЕТ О ВЫПОЛНЕНИИ ЛАБОРАТОРНОЙ РАБОТЫ №6  
«ПРОЦЕССЫ. РАБОТА С ПРОЦЕССАМИ.»  
по дисциплине  
«ОПЕРАЦИОННЫЕ СИСТЕМЫ»**

Выполнила,  
студентка группы МО32 \_\_\_\_\_ С.Н. Чупрова  
(подпись, дата)

Направление подготовки 02.03.03 Математическое обеспечение и  
администрирование информационных систем  
Курс 3

Отчет принял,  
преподаватель кафедры ИТ, доцент \_\_\_\_\_ А.А. Полупанов  
(подпись, дата)

Краснодар  
2025

## Процессы в Linux

Для просмотра таблицы процессов в Linux предназначена утилита *ps*.

Один из наиболее часто используемых ключей **aux**:

```
sa@astra:~$ ps aux | head -20
USER      PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root       1  0.0  0.5 103092 10812 ?        Ss  21:44  0:00 /sbin/init
root       2  0.0  0.0     0     0 ?        S   21:44  0:00 [kthreadd]
root       3  0.0  0.0     0     0 ?        I<  21:44  0:00 [rcu_gp]
root       4  0.0  0.0     0     0 ?        I<  21:44  0:00 [rcu_par_gp]
root       6  0.0  0.0     0     0 ?        I<  21:44  0:00 [kworker/0:0H-events_highpri]
root       9  0.0  0.0     0     0 ?        I<  21:44  0:00 [mm_percpu_wq]
root      10  0.0  0.0     0     0 ?        S   21:44  0:00 [rcu_tasks_rude_]
root      11  0.0  0.0     0     0 ?        S   21:44  0:00 [rcu_tasks_trace]
root      12  0.0  0.0     0     0 ?        S   21:44  0:00 [ksoftirqd/0]
root      13  0.0  0.0     0     0 ?        I   21:44  0:00 [rcu_sched]
root      14  0.0  0.0     0     0 ?        S   21:44  0:00 [migration/0]
root      15  0.0  0.0     0     0 ?        S   21:44  0:00 [idle_inject/0]
root      16  0.0  0.0     0     0 ?        S   21:44  0:00 [cpuhp/0]
root      17  0.0  0.0     0     0 ?        S   21:44  0:00 [kdevtmpfs]
root      18  0.0  0.0     0     0 ?        I<  21:44  0:00 [netns]
root      19  0.0  0.0     0     0 ?        I<  21:44  0:00 [inet_frag_wq]
root      20  0.0  0.0     0     0 ?        S   21:44  0:00 [kauditfd]
root      21  0.0  0.0     0     0 ?        S   21:44  0:00 [khungtaskd]
root      22  0.0  0.0     0     0 ?        S   21:44  0:00 [oom_reaper]
sa@astra:~$
```

Рисунок 1 – Просмотр процессов

Вывод процессов с идентификатором родителя:

```
sa@astra:~$ ps -ef | head -20
UID      PID  PPID  C STIME TTY      TIME CMD
root      1      0  0 дек16 ?      00:00:00 /sbin/init
root      2      0  0 дек16 ?      00:00:00 [kthreadd]
root      3      2  0 дек16 ?      00:00:00 [rcu_gp]
root      4      2  0 дек16 ?      00:00:00 [rcu_par_gp]
root      6      2  0 дек16 ?      00:00:00 [kworker/0:0H-events_highpri]
root      9      2  0 дек16 ?      00:00:00 [mm_percpu_wq]
root     10      2  0 дек16 ?      00:00:00 [rcu_tasks_rude_]
root     11      2  0 дек16 ?      00:00:00 [rcu_tasks_trace]
root     12      2  0 дек16 ?      00:00:00 [ksoftirqd/0]
root     13      2  0 дек16 ?      00:00:00 [rcu_sched]
root     14      2  0 дек16 ?      00:00:00 [migration/0]
root     15      2  0 дек16 ?      00:00:00 [idle_inject/0]
root     16      2  0 дек16 ?      00:00:00 [cpuhp/0]
root     17      2  0 дек16 ?      00:00:00 [kdevtmpfs]
root     18      2  0 дек16 ?      00:00:00 [netns]
root     19      2  0 дек16 ?      00:00:00 [inet_frag_wq]
root     20      2  0 дек16 ?      00:00:00 [kauditfd]
root     21      2  0 дек16 ?      00:00:00 [khungtaskd]
root     22      2  0 дек16 ?      00:00:00 [oom_reaper]
sa@astra:~$
```

Рисунок 2 – Вывод с ключом **-ef**

Используя значение PPID, можно легко найти все процессы, запущенные из текущей оболочки. Можно воспользоваться специальной утилитой pgrep и системной переменной \$\$, в которой содержится идентификатор текущего процесса:

```
sa@astra:~$ ps -f -p $$  
UID      PID  PPID  C STIME TTY          TIME CMD  
sa        8442  8439  0 дек16 pts/0  00:00:00 /bin/bash  
sa@astra:~$
```

Рисунок 3 – Все процессы, запущенные из текущей оболочки

С помощью утилиты pstree можно вывести список всех ПОТОМКОВ процесса с PID=0, которые были порождены ядром системы:

```
sa@astra:~$ pstree -p 0 | head -30  
?()--kthreadd(2)--acpi_thermal_pm(88)  
|   |-ata_sff(76)  
|   |-audit_prune_tre(364)  
|   |-blkcg_punt_bio(74)  
|   |-card0-crtc0(170)  
|   |-card0-crtc1(171)  
|   |-card0-crtc2(172)  
|   |-card0-crtc3(173)  
|   |-card0-crtc4(174)  
|   |-card0-crtc5(175)  
|   |-card0-crtc6(176)  
|   |-card0-crtc7(178)  
|   |-charger_manager(122)  
|   |-cpuhp/0(16)  
|   |-cryptd(381)  
|   |-devfreq_wq(79)  
|   |-ecryptfs-kthrea(85)  
|   |-edac-poller(78)  
|   |-ext4-rsv-conver(216)  
|   |-idle_inject/0(15)  
|   |-inet_frag_wq(19)  
|   |-iprt-VBoxWQueue(371)  
|   |-ipv6_addrconf(100)  
|   |-irq/18-vmwgfx(169)  
|   |-jbd2/sda1-8(215)  
|   |-kauditdd(20)  
|   |-kblockd(73)  
|   |-kcompactd0(24)  
|   |-kdevtmpfs(17)  
|   |-khugepaged(26)  
sa@astra:~$
```

Рисунок 4 – Вывод утилиты pstree

## Адресное пространство

Обычно процессы могут аллоцировать весь доступный объем памяти, и ничего настраивать дополнительного не требуется.

```
sa@astra:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e) 0
file size               (blocks, -f) unlimited
pending signals          (-i) 7524
max locked memory       (kbytes, -l) 65536
max memory size         (kbytes, -m) unlimited
open files              (-n) 1024
pipe size               (512 bytes, -p) 8
POSIX message queues    (bytes, -q) 819200
real-time priority      (-r) 0
stack size              (kbytes, -s) 8192
cpu time                (seconds, -t) unlimited
max user processes       (-u) 7524
virtual memory           (kbytes, -v) unlimited
file locks              (-x) unlimited
sa@astra:~$ █
```

Рисунок 5 – Значение, установленное для «virtual memory»

## Сигналы для процессов в Linux

Запускаем калькулятор в фоновом режиме:

```
| -khugepaged(26)
sa@astral:~$ ulimit -a
core file size          (blocks, -c) 0
data seg size           (kbytes, -d) unlimited
scheduling priority     (-e)
file size               (blocks, -f)
pending signals          (-i)
max locked memory       (kbytes, -l)
max memory size         (kbytes, -m)
open files              (-n)
pipe size               (512 bytes, -p)
POSIX message queues    (bytes, -q)
real-time priority      (-r)
stack size               (kbytes, -s)
cpu time                (seconds, -t)
max user processes       (-u)
virtual memory           (kbytes, -v)
file locks              (-x)
sa@astral:~$ kcalc &
[1] 8828
sa@astral:~$
```

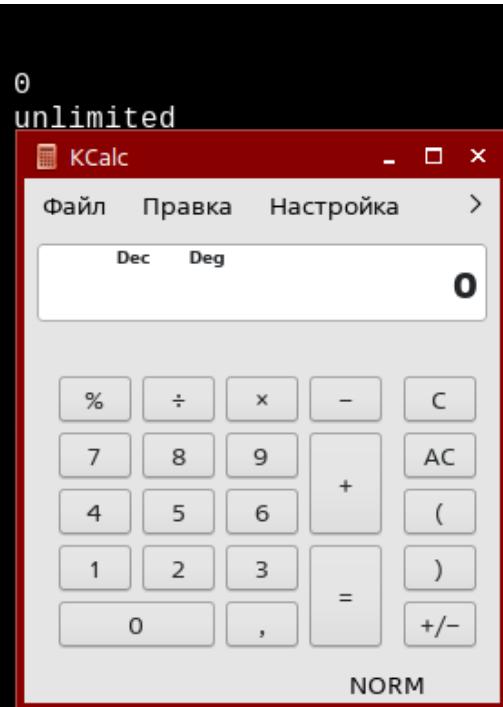


Рисунок 6 – Запуск процесса в фоновом режиме

Завершаем процесс с выбранным PID 8828 из прошлой команды:

```
[1]+ 8828
sa@astral:~$ kill -SIGTERM 8828
sa@astral:~$
```

Рисунок 7 – Завершение процесса

Процессы Linux поддерживают 64 сигнала, список которых можно посмотреть с помощью ключа -L (-l, --list) команды kill:

```
sa@astral:~$ kill -l
 1) SIGHUP      2) SIGINT      3) SIGQUIT      4) SIGILL      5) SIGTRAP
 6) SIGABRT     7) SIGBUS      8) SIGFPE      9) SIGKILL     10) SIGUSR1
 11) SIGSEGV    12) SIGUSR2     13) SIGPIPE     14) SIGALRM     15) SIGTERM
 16) SIGSTKFLT   17) SIGCHLD     18) SIGCONT     19) SIGSTOP     20) SIGTSTP
 21) SIGTTIN    22) SIGTTOU     23) SIGURG      24) SIGXCPU     25) SIGXFSZ
 26) SIGVTALRM   27) SIGPROF     28) SIGWINCH    29) SIGIO       30) SIGPWR
 31) SIGSYS     34) SIGRTMIN    35) SIGRTMIN+1  36) SIGRTMIN+2  37) SIGRTMIN+3
 38) SIGRTMIN+4  39) SIGRTMIN+5  40) SIGRTMIN+6  41) SIGRTMIN+7  42) SIGRTMIN+8
 43) SIGRTMIN+9  44) SIGRTMIN+10 45) SIGRTMIN+11 46) SIGRTMIN+12 47) SIGRTMIN+13
 48) SIGRTMIN+14 49) SIGRTMIN+15 50) SIGRTMAX-14 51) SIGRTMAX-13 52) SIGRTMAX-12
 53) SIGRTMAX-11 54) SIGRTMAX-10 55) SIGRTMAX-9  56) SIGRTMAX-8  57) SIGRTMAX-7
 58) SIGRTMAX-6  59) SIGRTMAX-5  60) SIGRTMAX-4  61) SIGRTMAX-3 62) SIGRTMAX-2
 63) SIGRTMAX-1  64) SIGRTMAX
[1]+  Завершено      kcalc
```

Рисунок 8 – Сигналы, поддерживаемые процессами Linux

Для того чтобы приложение игнорировало сигнал -1, его можно запустить с помощью команды nohup. Если закрыть терминал, такие процессы «осиротеют» и будут узочерены процессом init (systemd).

```
sa@astra:~$ nohup kcalc &
[1] 8884
sa@astra:~$ nohup: ввод игнорируется, вывод добавляется в 'nohup.out'
```

Рисунок 9 – Запуск команды через nohup

## **Планировщик задач в Linux и управление приоритетами процессов**

Группы процессов FIFO, RR и Other соответствуют политикам планирования SCHED\_FIFO, SCHED\_RR и SCHED\_OTHER (всего таких политик 6). Посмотреть список политик планирования можно командой chrt -m.

```
sa@astral:~$ chrt -m
SCHED_OTHER min/max priority : 0/0
SCHED_FIFO min/max priority  : 1/99
SCHED_RR min/max priority   : 1/99
SCHED_BATCH min/max priority : 0/0
SCHED_IDLE min/max priority : 0/0
SCHED_DEADLINE min/max priority : 0/0
sa@astral:~$ █
```

Рисунок 10 – Просмотр списка политик планирования

Просмотр фоновых заданий выполняется командой jobs.

```
sa@astral:~$ sleep 3000 &
[1] 8940
sa@astral:~$ sleep 3000 &
[2] 8941
sa@astral:~$ sleep 3000 &
[3] 8944
sa@astral:~$ jobs
[1]  Запущен      sleep 3000 &
[2]- Запущен      sleep 3000 &
[3]+ Запущен      sleep 3000 &
sa@astral:~$ █
```

Рисунок 11 – Список заданий

## Извлечение информации о процессах

Разберем, что хранится в каталогах /proc/PID/, где PID – числовой идентификатор процесса.

```
sa@astral:~$ sudo ls /proc/1
[sudo] пароль для sa:
arch_status cgroup      coredump_filter   environ    gid_map     map_files  mounts    numa_maps    pagemap    root      setgroups   stat      task      uid_map
attr        clear_refs  cpu_resctrl_groups  exe       io_maps    mountstats  oom_adj    patch_state sched     smaps    statm    timers_offsets wchan
autogroup   cmdline     cpuset           fd        limits    mem        net        oom_score   personality schedstat smaps_rollup status    timers
auxv       comm        cwd            fdinfo   loginuid  mountinfo  ns        oom_score_adj projid_map sessionid stack    syscall  timerslack_ns
sa@astral:~$
```

Рисунок 12 – Содержимое /proc/PID/

```
sa@astral:~$ cat /proc/1/cmdline && echo
/sbin/init
sa@astral:~$
```

Рисунок 13 – Стока запуска процесса

```
sa@astral:~$ sudo ls -l --color=always /proc/1/exe
lrwxrwxrwx 1 root root 0 дек  7 22:40 /proc/1/exe -> /usr/lib/systemd/systemd
```

Рисунок 14 – Символическая ссылка, ведущая к полному пути до исполняемого файла

```
sa@astral:~$ sudo ls -l --color=always /proc/1/cwd
lrwxrwxrwx 1 root root 0 дек 17 00:16 /proc/1/cwd -> /
```

Рисунок 15 – Текущий рабочий каталог процесса

```
sa@astral:~$ sudo cat /proc/1/environ && echo
SHLVL=1HOME=/init=/sbin/initTERM=linuxBOOT_IMAGE=/boot/vmlinuz-5.15.0-33-genericdrop_caps=PATH=/sbin:/usr/sbin:/bin:/usr/binPWD=/rootmnt=/root
sa@astral:~$
```

Рисунок 16 – Окружение процесса, создающее контекст его выполнения

```
sa@astras:~$ sudo ls -l /proc/1/fd --color=always | head -20
итого 0
lrwx----- 1 root root 64 дек  7 22:40 0 -> /dev/null
lrwx----- 1 root root 64 дек  7 22:40 1 -> /dev/null
lrwx----- 1 root root 64 дек 17 00:19 10 -> anon_inode:[eventpoll]
lrwx----- 1 root root 64 дек 17 00:19 100 -> socket:[19791]
lrwx----- 1 root root 64 дек 17 00:19 102 -> socket:[19638]
lrwx----- 1 root root 64 дек 17 00:19 104 -> socket:[19507]
lrwx----- 1 root root 64 дек 17 00:19 105 -> socket:[19371]
lrwx----- 1 root root 64 дек 17 00:19 108 -> socket:[17912]
lrwx----- 1 root root 64 дек 17 00:19 109 -> /dev/rfkill
lr-x----- 1 root root 64 дек 17 00:19 11 -> anon_inode:inotify
lrwx----- 1 root root 64 дек 17 00:19 111 -> socket:[16858]
lrwx----- 1 root root 64 дек 17 00:19 112 -> socket:[16860]
lr-x----- 1 root root 64 дек 17 00:19 113 -> pipe:[16856]
lrwx----- 1 root root 64 дек 17 00:19 114 -> /run/initctl
lrwx----- 1 root root 64 дек 17 00:19 115 -> socket:[16851]
lrwx----- 1 root root 64 дек 17 00:19 116 -> socket:[19106]
lrwx----- 1 root root 64 дек 17 00:19 117 -> socket:[16909]
lrwx----- 1 root root 64 дек 17 00:19 118 -> socket:[19104]
lrwx----- 1 root root 64 дек 17 00:19 119 -> socket:[16848]
sa@astras:~$
```

Рисунок 17 – Дескрипторы открытых файлов

```
sa@astras:~$ sudo cat /proc/1/io
rchar: 1989043514
wchar: 89884046
syscr: 1277284
syscw: 1385757
read_bytes: 1765844992
write_bytes: 35336192
cancelled_write_bytes: 11878400
sa@astras:~$
```

Рисунок 18 – Сведения об объемах данных, прочитанных и записанных  
процессом в хранилище информации

```

sa@astra:~$ sudo cat /proc/1/limits
Limit          Soft Limit      Hard Limit      Units
Max cpu time   unlimited     unlimited     seconds
Max file size  unlimited     unlimited     bytes
Max data size  unlimited     unlimited     bytes
Max stack size 8388608     unlimited     bytes
Max core file size 0           unlimited     bytes
Max resident set  unlimited     unlimited     bytes
Max processes   7524          7524        processes
Max open files  1048576       1048576     files
Max locked memory 67108864     67108864     bytes
Max address space  unlimited     unlimited     bytes
Max file locks  unlimited     unlimited     locks
Max pending signals 7524          7524        signals
Max msgqueue size 819200       819200      bytes
Max nice priority 0           0             0
Max realtime priority 0           0             0
Max realtime timeout  unlimited     unlimited     us
sa@astra:~$ █

```

Рисунок 19 – Ограничения процесса, установленные конфигурационным файлом

```

sa@astra:~$ sudo cat /proc/1/maps | head
5acdffb0f000-5acdffb3d000 r--p 00000000 08:01 396408
5acdffb3d000-5acdffc058000 r-xp 0002e000 08:01 396408
5acdffc058000-5acdffc0ad000 r--p 00149000 08:01 396408
5acdffc0ad000-5acdffc0e6000 r--p 0019d000 08:01 396408
5acdffc0e6000-5acdffc0e7000 rw-p 001d6000 08:01 396408
5acdffd77000-5acdffd15000 rw-p 00000000 00:00 0
[heap]
76ddc0000000-76ddc0021000 rw-p 00000000 00:00 0
76ddc0021000-76ddc4000000 ---p 00000000 00:00 0
76ddc793d000-76ddc793e000 ---p 00000000 00:00 0
76ddc793e000-76ddc813e000 rw-p 00000000 00:00 0
sa@astra:~$ █

```

Рисунок 20 – Физические адреса страниц памяти, используемые в данный момент

```
sa@astral:~$ sudo cat /proc/1/sched
systemd (1, #threads: 1)

-----  
se.exec_start : 9452281.348054  
se.vruntime : 711.435562  
se.sum_exec_runtime : 853.181618  
se.nr_migrations : 0  
nr_switches : 5616  
nr_voluntary_switches : 1956  
nr_involuntary_switches : 3660  
se.load.weight : 1048576  
se.avg.load_sum : 149  
se.avg.runnable_sum : 152576  
se.avg.util_sum : 103424  
se.avg.load_avg : 0  
se.avg.runnable_avg : 0  
se.avg.util_avg : 0  
se.avg.last_update_time : 9452281347072  
se.avg.util_est.ewma : 8  
se.avg.util_est.enqueued : 0  
uclamp.min : 0  
uclamp.max : 1024  
effective_uclamp.min : 0  
effective_uclamp.max : 1024  
policy : 0  
prio : 120  
clock-delta : 14  
mm->numa_scan_seq : 0  
numa_pages_migrated : 0  
numa_preferred_nid : -1  
total_numa_faults : 0  
current_node=0, numa_group_id=0  
numa_faults node=0 task_private=0 task_shared=0 group_private=0 group_shared=0  
sa@astral:~$
```

Рисунок 21 – Текущие значения переменных планировщика процессов

```
sa@astral:~$ sudo cat /proc/1/stat
(systemd) S 0 1 0 -1 4194560 19145 679562 171 33897 24 60 2427 2437 28 0 1 0 4 105566208 2688 18446744073709551615 99840741855232 99840743014265 140736789485168 0 0 671173123 4096 1
2688 0 17 0 0 0 0 0 99840743366352 99840743596352 99840773550888 140736789487435 140736789487446 140736789487597 8
sa@astral:~$
```

Рисунок 22 – Основные сведения о процессе в машиночитаемом формате

```
sa@astral:~$ sudo cat /proc/1/status | head -20
Name:      systemd
Umask:    0000
State:    S (sleeping)
Tgid:     1
Ngid:     0
Pid:      1
PPid:    0
TracerPid: 0
Uid:      0      0      0      0
Gid:      0      0      0      0
FDSize:   128
Groups:
NSgid:   1
NSpid:   1
NSpgid:  1
NSsid:   1
VmPeak:   167636 kB
VmSize:   103092 kB
VmLck:    0 kB
VmPin:    0 kB
sa@astral:~$
```

Рисунок 23 – Основные сведения о процессе в человекочитаемом формате

```
sa@astral:~$ cat /proc/1/statm
25773 2688 2004 283 0 4905 0
sa@astral:~$
```

Рисунок 24 – Статистика по использованию памяти

## Содержимое /proc

```
sasha@astral:~$ ls /proc
```

cpuinfo	fs	kpagegroup	mtrr	stat	version_signature
170	84	8876	91	98	
122	20	7784	92	992	crypto
122	278	8409	93	992	interrupts
13	21	3213	94	992	kpagecount
13	287	3213	94	992	net
13	287	3189	94	992	swaps
131	21	3216	94	992	vmallocinfo
14	215	3217	94	992	vmstat
1486	216	3218	94	992	devices
1486	2955	3218	94	992	iomem
15	22	3222	94	992	kpaflags
15	22	3232	94	992	pagetypeinfo
15	22	3243	94	992	sys
15	22	3255	94	992	partition
15	22	3255	94	992	sysrq-trigger
15	22	3255	94	992	zoneinfo
16	23	3266	94	992	irq
16	23	3266	94	992	locks
16	23	3266	94	992	pressure
16	23	3266	94	992	sysvipc
16	23	3266	94	992	thread-self
16	23	3266	94	992	timer_list
163	178	3272	94	992	bus
163	178	3272	94	992	dynamic_debug
164	169	3278	94	992	kcORE
164	169	3278	94	992	meminfo
17	19	3280	94	992	scsi
17	19	3280	94	992	self
17	19	3280	94	992	tty
17	19	3280	94	992	cmdline
17	19	3280	94	992	fb
17	19	3280	94	992	key-users
17	19	3280	94	992	modules
17	19	3280	94	992	slabinfo
17	19	3280	94	992	uptime
17	19	3280	94	992	filesystems
17	19	3280	94	992	kmsg
17	19	3280	94	992	mounts
17	19	3280	94	992	softirqs
17	19	3280	94	992	version

```
sasha@astral:~$
```

### Рисунок 25 – Содержимое каталога /proc

```
sa@astral:~$ cat /proc/cmdline
BOOT_IMAGE=/boot/vmlinuz-5.15.0-33-generic root=UUID=eba34003-adda-47f3-a50d-8b9513eb81dc ro parsec.max_ilev=63 quiet net.ifnames=0
sa@astral:~$
```

Рисунок 26 – Список параметров, которые были переданы ядру при загрузке

```
sa@astral:~$ cat /proc/cpuinfo | head -20
processor       : 0
vendor_id      : GenuineIntel
cpu family     : 6
model          : 151
model name     : 12th Gen Intel(R) Core(TM) i7-12700F
stepping        : 2
cpu MHz        : 2023.262
cache size     : 25600 KB
physical id    : 0
siblings        : 1
core id        : 0
cpu cores      : 1
apicid          : 0
initial apicid : 0
fpu             : yes
fpu_exception   : yes
cpuid level    : 22
wp              : yes
flags           : fpu vme de pse tsc msr pae mce cx8 apic sep mtrr pge mca cmov pat pse36 clflush mmx fxsr_sse sse2 ht syscall nx rdtscp lm constant_tsc rep_good nopl xtopology nonstop_ts c
cpuid pn1_pml1ndq monitor asse1 cx16 pcid sse4_1 sse4_2 x2apic movbe pcnpct aes xsave avx rdrand lahf_lm abm 3unowprefetch invpcid_single ibrs_enhanced fsbsbase avx2 invpcid rdseed cl
flushopt md_clear flush_l1d arch_capabilities
bugs            : spectre_v1 spectre_v2 spec_store_bypass swaps
sa@astral:~$
```

Рисунок 27 – Сведения о всех установленных процессорах

Рисунок 28 – Статистика операций со всеми дисками

Файл /proc/meminfo – отображение информации о состоянии памяти.

Предоставляет больше параметров, чем утилита free.

```
sa@astra:~$ free
              total        used        free      shared  buff/cache   available
Mem:       2025456     824760     113264      70836     1087432      952448
Swap:      998396      77628      920768
sa@astra:~$
```

Рисунок 29 – Информация, предоставленная утилитой free

```
sa@astra:~$ cat /proc/meminfo | head -20
MemTotal:          2025456 kB
MemFree:           82520 kB
MemAvailable:      921876 kB
Buffers:           112888 kB
Cached:            882584 kB
SwapCached:        1204 kB
Active:             514844 kB
Inactive:          1150956 kB
Active(anon):      36892 kB
Inactive(anon):    704272 kB
Active(file):      477952 kB
Inactive(file):    446684 kB
Unevictable:        0 kB
Mlocked:            0 kB
SwapTotal:          998396 kB
SwapFree:           920768 kB
Dirty:              60 kB
Writeback:           0 kB
AnonPages:          669312 kB
Mapped:             337204 kB
sa@astra:~$ █
```

Рисунок 30 – Информация, предоставленная файлом /proc/meminfo

```
sa@astra:~$ cat /proc/devices | head -20
Character devices:
  1 mem
  4 /dev/vc/0
  4 tty
  4 ttys
  5 /dev/tty
  5 /dev/console
  5 /dev/ptmx
  5 ttyprintk
  6 lp
  7 vcs
 10 misc
 13 input
 21 sg
 29 fb
 89 i2c
 99 ppdev
108 ppp
116 alsa
128 ptm
sa@astra:~$ █
```

Рисунок 31 – Перечень устройств в системе

```
sa@astra:~$ cat /proc/filesystems
nodev    sysfs
nodev    tmpfs
nodev    bdev
nodev    proc
nodev    cgroup
nodev    cgroup2
nodev    cpuset
nodev    devtmpfs
nodev    configfs
nodev    debugfs
nodev    tracefs
nodev    securityfs
nodev    sockfs
nodev    bpf
nodev    pipefs
nodev    ramfs
nodev    hugetlbfs
nodev    devpts
        ext3
        ext2
        ext4
        squashfs
        vfat
nodev    ecryptfs
        fuseblk
nodev    fuse
nodev    fusectl
nodev    mqueue
nodev    pstore
nodev    parsecfs
nodev    autofs
nodev    binfmt_misc
sa@astra:~$ █
```

Рисунок 32 – Перечень файловых систем, поддерживаемых ядром ОС

```

sa@astral:~$ cat /proc/mounts | head -20
sysfs /sys sysfs rw,nosuid,nodev,noexec,relatime 0 0
proc /proc proc rw,nosuid,nodev,noexec,relatime 0 0
udev /dev devtmpfs rw,nosuid,relatime,size=963204k,nr_inodes=240801,mode=755,inode64 0 0
devpts /dev/pts devpts rw,nosuid,noexec,relatime,gid=5,mode=620,ptmxmode=000 0 0
tmpfs /run tmpfs rw,nosuid,noexec,relatime,size=202548k,mode=755,inode64 0 0
/dev/sda1 / ext4 rw,relatime,errors=remount-ro 0 0
parsecfs /parsecfs parsecfs rw,sync,relatime 0 0
securityfs /sys/kernel/security securityfs rw,nosuid,nodev,noexec,relatime 0 0
tmpfs /dev/shm tmpfs rw,nosuid,nodev,inode64 0 0
tmpfs /run/lock tmpfs rw,nosuid,nodev,noexec,relatime,size=5120k,inode64 0 0
tmpfs /sys/fs/cgroup tmpfs ro,nosuid,nodev,noexec,mode=755,inode64 0 0
cgroup2 /sys/fs/cgroup/unified cgroup2 rw,nosuid,nodev,noexec,relatime,nsdelegate 0 0
cgroup /sys/fs/cgroup/systemd cgroup rw,nosuid,nodev,noexec,relatime,xattr,name=systemd 0 0
pstore /sys/fs/pstore pstore rw,nosuid,nodev,noexec,relatime 0 0
bpf /sys/fs/bpf bpf rw,nosuid,nodev,noexec,relatime,mode=700 0 0
cgroup /sys/fs/cgroup/devices cgroup rw,nosuid,nodev,noexec,relatime,devices 0 0
cgroup /sys/fs/cgroup/freezer cgroup rw,nosuid,nodev,noexec,relatime,freezer 0 0
cgroup /sys/fs/cgroup/hugetlb cgroup rw,nosuid,nodev,noexec,relatime,hugetlb 0 0
cgroup /sys/fs/cgroup/cpu,cpuacct cgroup rw,nosuid,nodev,noexec,relatime,cpu,cpuacct 0 0
cgroup /sys/fs/cgroup/rdma cgroup rw,nosuid,nodev,noexec,relatime,rdma 0 0
sa@astral:~$ 

```

Рисунок 33 – Перечень смонтированных файловых систем

```

sa@astral:~$ cat /proc/modules | head -20
binfmt_misc 24576 1 - Live 0x0000000000000000 (E)
vboxvideo 36864 0 - Live 0x0000000000000000 (0E)
drm_ttm_helper 16384 1 vboxvideo, Live 0x0000000000000000 (E)
intel_rapl_msr 20480 0 - Live 0x0000000000000000 (E)
joydev 32768 0 - Live 0x0000000000000000 (E)
intel_rapl_common 32768 1 intel_rapl_msr, Live 0x0000000000000000 (E)
intel_powerclamp 20480 0 - Live 0x0000000000000000 (E)
crct10dif_pclmul 16384 1 - Live 0x0000000000000000 (E)
crc32_pclmul 16384 0 - Live 0x0000000000000000 (E)
ghash_clmulni_intel 16384 0 - Live 0x0000000000000000 (E)
aesni_intel 376832 0 - Live 0x0000000000000000 (E)
crypto_simd 16384 1 aesni_intel, Live 0x0000000000000000 (E)
cryptd 24576 2 ghash_clmulni_intel,crypto_simd, Live 0x0000000000000000 (E)
rapl 20480 0 - Live 0x0000000000000000 (E)
intel_cstate 20480 0 - Live 0x0000000000000000 (E)
snd_intel8x0 49152 3 - Live 0x0000000000000000 (E)
input_leds 16384 0 - Live 0x0000000000000000 (E)
snd_ac97_codec 155648 1 snd_intel8x0, Live 0x0000000000000000 (E)
serio_raw 20480 0 - Live 0x0000000000000000 (E)
ac97_bus 16384 1 snd_ac97_codec, Live 0x0000000000000000 (E)
sa@astral:~$ 

```

Рисунок 34 – Список подгруженных модулей ядра

Filename	Type	Size	Used	Priority
/dev/sda5	partition	998396	88932	-2

Рисунок 35 – Список разделов подкачки

```

sa@astral:~$ cat /proc/version
Linux version 5.15.0-33-generic (builder@build5) (gcc (AstraLinuxSE 8.3.0-6) 8.3.0, GNU ld (GNU Binutils for AstraLinux) 2.31.1) #astra2+ci96 SMP Fri Oct 28 18:23:37 UTC 2022
sa@astral:~$ 

```

Рисунок 36 – Версия ядра ОС

Каталог /sys/kernel/ содержит набор файлов, которые позволяют нам оперативно без перезагрузки изменять параметры ядра ОС:

```
sa@astra:~$ ls /proc/sys/kernel/ | head -20
acct
acpi_video_flags
auto_msgmni
bootloader_type
bootloader_version
bpf_stats_enabled
cad_pid
cap_last_cap
core_pattern
core_pipe_limit
core_uses_pid
ctrl-alt-del
dmesg_restrict
domainname
firmware_config
ftrace_dump_on_oops
ftrace_enabled
hardlockup_all_cpu_backtrace
hardlockup_panic
hostname
sa@astra:~$ █
```

Рисунок 37 – Содержимое каталога /sys/kernel/

## Управление процессами

Для управления процессами в Linux существует набор утилит. Рассмотрим работу с основными из них: консольными утилитами (ps, top и htop, kill):

```
sa@astra:~$ ps
  PID TTY      TIME CMD
 8442 pts/0    00:00:00 bash
 8940 pts/0    00:00:00 sleep
 8941 pts/0    00:00:00 sleep
 8944 pts/0    00:00:00 sleep
 9280 pts/0    00:00:00 ps
sa@astra:~$
```

Рисунок 38 – Просмотр процессов через утилиту ps

```
sa@astra:~$ ps aux --sort=%mem | tail -n 8
sa          3649  2.4  0.9 247816 141508 ?        Ssl  дек16  3:37 /usr/lib/firefox/firefox -contentproc -childID 6 -isForBrowser -prefsLen 27113 -prefMapSize 224567 -jsInitLen 247228 -pare
fly-dm   2988  0.5 13.4 794588 272836 tty7  Ssl+  дек16  0:57 /usr/lib/xorg/Xorg -br -novtswitch -quiet -keeptty :0 vt7 -logfile /var/log/fly-dm/Xorg.%s.log -seat seat0 -auth /var/run/
xauth/A:0-Esgyxb
sa          3364  1.3 16.9 3079396 343672 ?        Ssl  дек16  2:14 /usr/lib/firefox/firefox
sa@astra:~$
```

Рисунок 39 – Сортировка процессов

```
sa@astra:~$ ps -eo euser,ruser,suser,fuser,f,comm,label
EUSER   RUSER   SUSER   FUSER   F COMMAND           LABEL
root    root    root    root    4 systemd          0:63:0:0
root    root    root    root    1 kthreadd         0:63:0:0
root    root    root    root    1 rcu_gp          0:63:0:0
root    root    root    root    1 rcu_par_gp       0:63:0:0
root    root    root    root    1 kworker/0:0H-ev   0:63:0:0
root    root    root    root    1 mm_percpu_wq     0:63:0:0
root    root    root    root    1 rcu_tasks_rude_  0:63:0:0
root    root    root    root    1 rcu_tasks_trace  0:63:0:0
root    root    root    root    1 ksoftirqd/0      0:63:0:0
root    root    root    root    1 rcu_sched         0:63:0:0
root    root    root    root    1 migration/0      0:63:0:0
root    root    root    root    1 idle_inject/0    0:63:0:0
root    root    root    root    1 cpuhp/0          0:63:0:0
root    root    root    root    5 kdevtmpfs        0:63:0:0
root    root    root    root    1 netns            0:63:0:0
root    root    root    root    1 inet_frag_wq     0:63:0:0
root    root    root    root    1 kaudited         0:63:0:0
root    root    root    root    1 khungtaskd       0:63:0:0
root    root    root    root    1 oom_reaper        0:63:0:0
root    root    root    root    1 writeback         0:63:0:0
root    root    root    root    1 kcompactd0       0:63:0:0
root    root    root    root    1 ksmd             0:63:0:0
root    root    root    root    1 khugepaged       0:63:0:0
root    root    root    root    1 integrityd        0:63:0:0
root    root    root    root    1 kblockd          0:63:0:0
root    root    root    root    1 blkcg_punt_bio  0:63:0:0
root    root    root    root    1 tpm_dev_wq       0:63:0:0
root    root    root    root    1 ata_sff          0:63:0:0
root    root    root    root    1 md               0:63:0:0
root    root    root    root    1 edac-poller       0:63:0:0
root    root    root    root    1 devfreq_wq       0:63:0:0
root    root    root    root    1 watchdogd        0:63:0:0
root    root    root    root    1 kworker/0:1H-kb   0:63:0:0
root    root    root    root    1 kswapd0          0:63:0:0
```

Рисунок 40 – Информация об атрибутах EUID, RUID, SUID.

```

sa@astra:~$ ps -eLf | head -20
UID      PID  PPID  LWP   C NLWP STIME TTY          TIME CMD
root      1    0     1  0   1 дек16 ?        00:00:00 /sbin/init
root      2    0     2  0   1 дек16 ?        00:00:00 [kthreadd]
root      3    2     3  0   1 дек16 ?        00:00:00 [rcu_gp]
root      4    2     4  0   1 дек16 ?        00:00:00 [rcu_par_gp]
root      6    2     6  0   1 дек16 ?        00:00:00 [kworker/0:0H-events_highpri]
root      9    2     9  0   1 дек16 ?        00:00:00 [mm_percpu_wq]
root     10    2    10  0   1 дек16 ?        00:00:00 [rcu_tasks_rude_]
root     11    2    11  0   1 дек16 ?        00:00:00 [rcu_tasks_trace]
root     12    2    12  0   1 дек16 ?        00:00:00 [ksoftirqd/0]
root     13    2    13  0   1 дек16 ?        00:00:01 [rcu_sched]
root     14    2    14  0   1 дек16 ?        00:00:00 [migration/0]
root     15    2    15  0   1 дек16 ?        00:00:00 [idle_inject/0]
root     16    2    16  0   1 дек16 ?        00:00:00 [cpuhp/0]
root     17    2    17  0   1 дек16 ?        00:00:00 [kdevtmpfs]
root     18    2    18  0   1 дек16 ?        00:00:00 [netns]
root     19    2    19  0   1 дек16 ?        00:00:00 [inet_frag_wq]
root     20    2    20  0   1 дек16 ?        00:00:00 [kaudittd]
root     21    2    21  0   1 дек16 ?        00:00:00 [khungtaskd]
root     22    2    22  0   1 дек16 ?        00:00:00 [oom_reaper]
sa@astra:~$ █

```

Рисунок 41 – Просмотр потоков с помощью команды ps -eLf

```

sa@astra:~$ ps axmu | head -20
USER      PID %CPU %MEM     VSZ   RSS TTY      STAT START   TIME COMMAND
root      1  0.0  0.5 103092 10708 ?
root      -  0.0  -
root      2  0.0  0.0     0   0 ?       -  дек16  0:00 /sbin/init
root      -  0.0  -
root      3  0.0  0.0     0   0 ?       -  дек16  0:00 [kthreadd]
root      -  0.0  -
root      4  0.0  0.0     0   0 ?       -  дек16  0:00 [rcu_gp]
root      -  0.0  -
root      6  0.0  0.0     0   0 ?       -  дек16  0:00 [kworker/0:0H-events_highpri]
root      -  0.0  -
root      9  0.0  0.0     0   0 ?       -  дек16  0:00 [mm_percpu_wq]
root      -  0.0  -
root     10  0.0  0.0     0   0 ?       -  дек16  0:00 [rcu_tasks_rude_]
root      -  0.0  -
root     11  0.0  0.0     0   0 ?       -  дек16  0:00 [rcu_tasks_trace]
root      -  0.0  -
root     12  0.0  0.0     0   0 ?       -  дек16  0:00 [ksoftirqd/0]
root      -  0.0  -
root     13  0.0  0.0     0   0 ?       -  дек16  0:01 [rcu_sched]
sa@astra:~$ █

```

Рисунок 42 – Просмотр потоков с помощью команды ps axmu

```
sa@astral:~$ pstree | head -20
systemd--NetworkManager---dhclient---{dhclient}
|                               `-{NetworkManager}[]
|-2*[VBoxClient---VBoxClient---3*[{VBoxClient}]]]
|-VBoxClient---VBoxClient
|-VBoxDRMClient---4*[{VBoxDRMClient}]
|-VBoxService---8*[{VBoxService}]
|-agetty
|-alsactl
|-astra-orientati---2*[{astra-orientati}]
|-at-spi2-registr---2*[{at-spi2-registr}]
|-auditd---{auditd}
|-avahi-daemon---avahi-daemon
|-cron
|-cupsd
|-2*[dbus-daemon]
|-dbus-launch
|-fly-baloorunner---2*[{fly-baloorunner}]
|-fly-dm---Xorg---5*[{Xorg}]
|           `---fly-dm---fly-wm---astra-event-wat---{astra-event-wat}
|                           |-at-spi-bus-laun---dbus-daemon
sa@astral:~$ █
```

Рисунок 43 – Дерево процессов

```
sa@astral:~$ pstree -p | head -20
systemd(1)-+NetworkManager(395)-+dhclient(8130)-+{dhclient}(8136)
|                               |-{NetworkManager}(436)
|                               `-{NetworkManager}(438)
|-VBoxClient(3171)-+VBoxClient(3172)-+{VBoxClient}(3173)
|                               |-{VBoxClient}(3174)
|                               `-{VBoxClient}(3175)
|-VBoxClient(3180)-+VBoxClient(3181)
|-VBoxClient(3185)-+VBoxClient(3186)-+{VBoxClient}(3187)
|                               |-{VBoxClient}(3188)
|                               `-{VBoxClient}(3189)
|-VBoxDRMClient(912)-+{VBoxDRMClient}(916)
|                               |-{VBoxDRMClient}(917)
|                               |-{VBoxDRMClient}(918)
|                               `-{VBoxDRMClient}(8392)
|-VBoxService(920)-+{VBoxService}(921)
|                               |-{VBoxService}(922)
|                               |-{VBoxService}(923)
|                               |-{VBoxService}(924)
|                               |-{VBoxService}(925)
|                               `-{VBoxService}(926)
sa@astral:~$ █
```

Рисунок 44 – Вывод PID процессов

```
|          | -{VBoxSer
sa@astra:~$ ps -1
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss      0:00 /sbin/init
sa@astra:~$ ps 1
  PID TTY      STAT   TIME COMMAND
    1 ?        Ss      0:00 /sbin/init
sa@astra:~$ ps -p "1 2"
  PID TTY          TIME CMD
    1 ?      00:00:00 systemd
    2 ?      00:00:00 kthreadd
sa@astra:~$ █
```

Рисунок 45 – Фильтрация вывода команды ps

```
sa@astra:~$ ps -C bash
  PID TTY          TIME CMD
  8442 pts/0      00:00:00 bash
sa@astra:~$ █
```

Рисунок 46 – Вывод информации по процессам

```
sa@astra:~$ ps -C bash -o pid
  PID
  8442
sa@astra:~$ ps -C bash -o pid=
  8442
sa@astra:~$ █
```

Рисунок 47 – Вывод только колонки с PID найденных процессов и только PID  
без названия колонки

```
sa@astra:~$ ps aux | grep tty
root     840  0.0  0.0  5808 1740  ttys0  Ss+  дек16  0:00 /sbin/agetty -o -p -- \u --noclear ttys0 linux
fly-dm  2988  0.5 13.4 794508 272836  ttys1  Ssl+  дек16  1:00 /usr/lib/xorg/Xorg -br -novtswitch -quiet -keeptty :0 vt7 -logfile /var/log/fly-dm/Xorg.%s.log -seat seat0 -auth /var/run/
xauth/A:0-EsgYxb
sa      9474  0.0  0.0  6096  880  pts/0  S+   06:57  0:00 grep tty
sa@astra:~$ █
```

Рисунок 48 – Дополнительная фильтрация с помощью утилиты grep