Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ **ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ**

КАФЕДРА **КОМПЬЮТЕРНЫЕ СИСТЕМЫ И СЕТИ (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ 09.03.01 Информатика и вычислительная техника

О Т Ч Е Т

по лабораторной работе № 4

Название: Исследование способов организации оперативной памяти и взаимодействия процессов

Дисциплина: Операционные системы

Студент ИУ6-53Б _____ В.К. Залыгин
(Группа) (Подпись, дата) (И.О. Фамилия)

Преподаватель _____ В.Ю. Мельников
(Подпись, дата)
_____ А.М. Суровов
(И.О. Фамилия)

Москва, 2024

ВВЕДЕНИЕ

Цель лабораторной работы

Цель данной работы — получение теоретических и практических сведений об управлении процессами, потоками и оперативной памятью в UNIX-подобных системах и в Linux в частности.

Выполнение

• Открыть в текстовом браузере некую страницу и перевести его в фоновый режим

Сначала откроем в консольном браузере сайт и уведем его в фон.

Рисунок 1 – Результат перевода браузера в фоновый режим

• Запустить ещё два экземпляра текстового браузера в фоновом режиме

Откроем еще две страницы и уведем их в фон.

Рисунок 2 – Еще два запущенных в фоне процесса

• Найти процесс, максимально нагружающий процессор

Используем утилиту top и посмотрим самый ресурсозатратный процесс.

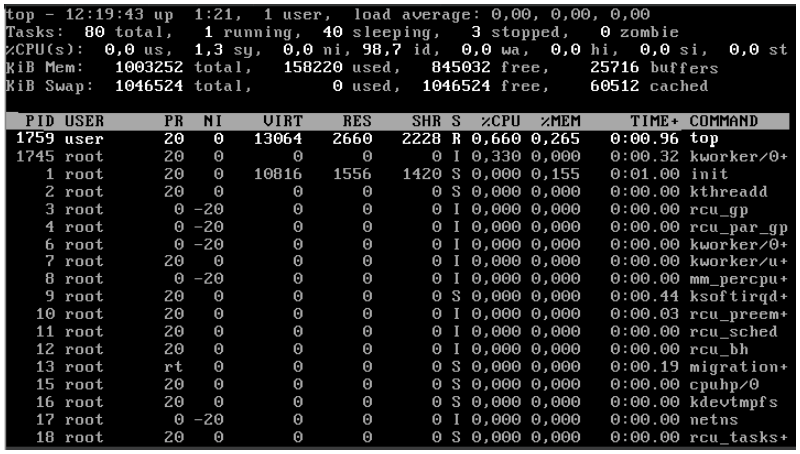

Рисунок 3 – Больше всего ресурсов CPU потребляет программа top

• «Убить» первый процесс браузера в котором открыта 1 страница

С помощью команды ps a посмотрим все процессы и найдем процессы браузеров. После чего завершим один из них принудительно.

```
[user@zalygin ~]$ ps a | grep w3m
 1747 tty1     T      0:00 w3m google.com
 1754 tty1     T      0:00 w3m ya.ru
 1755 tty1     T      0:00 w3m yandex.ru
 1766 tty1     S+     0:00 grep --color=auto w3m
[user@zalygin ~]$ kill 1747
[user@zalygin ~]$ ps a | grep w3m
 1747 tty1     T      0:00 w3m google.com
 1754 tty1     T      0:00 w3m ya.ru
 1755 tty1     T      0:00 w3m yandex.ru
 1768 tty1     S+     0:00 grep --color=auto w3m
[user@zalygin ~]$ kill -9 1747
[2]   ▪▪▪▪▪               w3m google.com
[user@zalygin ~]$ ps a | grep w3m
 1754 tty1     T      0:00 w3m ya.ru
 1755 tty1     T      0:00 w3m yandex.ru
 1770 tty1     S+     0:00 grep --color=auto w3m
[user@zalygin ~]$
```

Рисунок 4 – Принудительное завершение процесса первого браузера

• Вывести список всех процессов всех пользователей

Вывод всех процессов пользователей командой ps a.

```
root      1108  0.0  0.8 430640  8832 ?        Ssl  10:58   0:00 /usr/sbin/Modem
polkitd   1140  0.0  1.6 529588 16528 ?        Sl   10:58   0:00 /usr/libexec/po
root      1179  0.0  1.3 373128 13120 ?        Ssl  10:58   0:00 /usr/sbin/Netwo
root      1228  0.0  0.0      0     0 ?        I<   10:58   0:00 [ipv6_addrconf]
root      1234  0.0  0.3  45340  3788 ?        Ss   10:58   0:00 /usr/libexec/bl
syslogd   1369  0.0  0.1  12644  1572 ?        Ss   10:58   0:00 /sbin/syslogd -
klogd     1420  0.0  0.1   6340  1584 ?        Ss   10:58   0:00 /sbin/klogd -c
root      1525  0.0  0.1  19320  1956 ?        SNs  10:58   0:00 /usr/sbin/crond
_avahi    1579  0.0  0.3  51772  3536 ?        S    10:58   0:00 avahi-daemon: r
_avahi    1580  0.0  0.0  51644   368 ?        S    10:58   0:00 avahi-daemon: c
root      1631  0.0  0.0  49076   608 ?        Ss   10:58   0:00 /usr/sbin/sshd
root      1643  0.0  0.3  72292  3500 tty1     Ss   10:58   0:00 /bin/login
root      1644  0.0  0.0   4200   636 tty2     Ss+  10:58   0:00 /sbin/mingetty
root      1645  0.0  0.0   4200   736 tty3     Ss+  10:58   0:00 /sbin/mingetty
root      1646  0.0  0.0   4200   740 tty4     Ss+  10:58   0:00 /sbin/mingetty
root      1647  0.0  0.0   4200   640 tty5     Ss+  10:58   0:00 /sbin/mingetty
root      1648  0.0  0.0   4200   680 tty6     Ss+  10:58   0:00 /sbin/mingetty
user      1669  0.0  0.3  15716  3444 tty1     S    10:58   0:01 -bash
root      1745  0.1  0.0      0     0 ?        I    12:15   0:00 [kworker/0:1-ev
user      1754  0.0  0.5  50472  5720 tty1     T    12:17   0:00 w3m ya.ru
user      1755  0.0  0.5  50472  5684 tty1     T    12:18   0:00 w3m yandex.ru
root      1762  0.1  0.0      0     0 ?        I    12:20   0:00 [kworker/0:2-at
root      1771  0.1  0.0      0     0 ?        I    12:25   0:00 [kworker/0:0-ev
user      1772  0.0  0.2  10816  2420 tty1     R+   12:26   0:00 ps axu
[user@zalygin ~]$ _
```

Рисунок 5 – Вывод всех процессов всех пользователей

• Просмотреть список процессов постранично

Так как список выше, чем размер терминала, удобно будет использовать команду последовательной выдачи строк more.

```
USER       PID %CPU %MEM    VSZ   RSS TTY      STAT START   TIME COMMAND
root         1  0.0  0.1  10816  1556 ?        Ss   10:58   0:01 init [4]

root         2  0.0  0.0      0     0 ?        S    10:58   0:00 [kthreadd]
root         3  0.0  0.0      0     0 ?        I<   10:58   0:00 [rcu_gp]
root         4  0.0  0.0      0     0 ?        I<   10:58   0:00 [rcu_par_gp]
root         6  0.0  0.0      0     0 ?        I<   10:58   0:00 [kworker/0:0H-k
b]
root         7  0.0  0.0      0     0 ?        I    10:58   0:00 [kworker/u2:0-s
c]
root         8  0.0  0.0      0     0 ?        I<   10:58   0:00 [mm_percpu_wq]
root         9  0.0  0.0      0     0 ?        S    10:58   0:00 [ksoftirqd/0]
root        10  0.0  0.0      0     0 ?        I    10:58   0:00 [rcu_preempt]
root        11  0.0  0.0      0     0 ?        I    10:58   0:00 [rcu_sched]
root        12  0.0  0.0      0     0 ?        I    10:58   0:00 [rcu_bh]
root        13  0.0  0.0      0     0 ?        S    10:58   0:00 [migration/0]
root        15  0.0  0.0      0     0 ?        S    10:58   0:00 [cpuhp/0]
root        16  0.0  0.0      0     0 ?        S    10:58   0:00 [kdevtmpfs]
root        17  0.0  0.0      0     0 ?        I<   10:58   0:00 [netns]
root        18  0.0  0.0      0     0 ?        S    10:58   0:00 [rcu_tasks_kthr
e]
root        19  0.0  0.0      0     0 ?        S    10:58   0:00 [kauditd]
root        20  0.0  0.0      0     0 ?        S    10:58   0:00 [khungtaskd]
root        21  0.0  0.0      0     0 ?        S    10:58   0:00 [oom_reaper]
--▪▪▪▪▪--_
```

Рисунок 6 – Использование утилиты more для последовательной выдачи данных

• Отобрать из вывода команды ps строку, соответствующую процессу «dbus-daemon», определить, где лежит её выполняемый файл и с какими параметрами он запущен

Выведем строку, относящуюся к процессу dbus-daemon

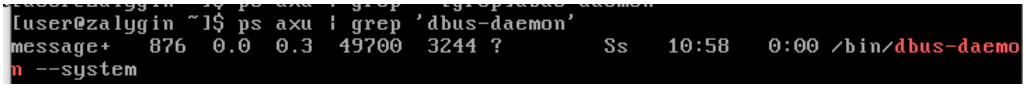

Рисунок 7 – Строка вывода ps, соответствующая процессу dbus-daemon

• Записать в файл с именем, содержащим текущее время, строку «-------»и список процессов

Чтобы создать файл с датой и требуемой строкой, можно использовать группу команд.

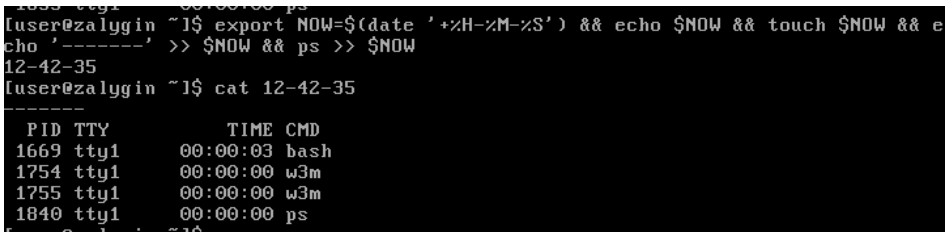

Рисунок 8 – Создание файла с текущем временем, запись в него строки и списка процессов

• Выполнить команду в фоновом режиме с отсрочкой запуска на 1 минуту. Продемонстрировать, что команда выполнилась именно через минуту.

Чтобы отсрочить команду на минуту, можно ее предварять командой date.

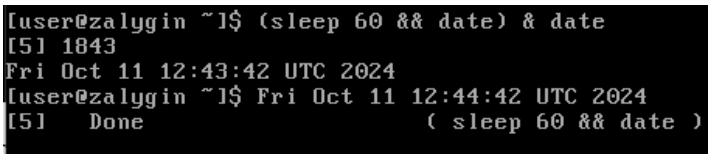

Рисунок 9 – Отсрочка команды на 1 минуту

• Отобрать из одного из сформированных файлов строки, относящиеся к одному из процессов.

Используем grep, чтобы вывести строки процессов, относящиеся к браузеру.

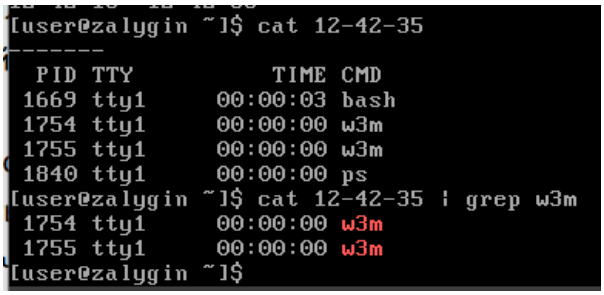

Рисунок 10 – Фильтрация строк, относящихся только к утилите w3m

• Вывести результаты работы произвольной команды в один файл, а сообщения об ошибках в другой. Продемонстрировать правильность работы.

Чтобы перенаправить потоки вывода и ошибок, нужно использовать специальные операторы >, 2>.

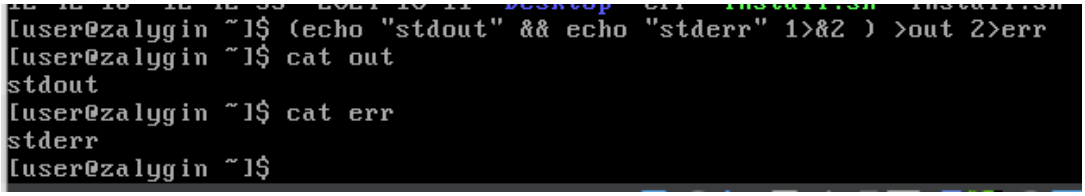

Рисунок 11 – Вывод потока вывода и потока ошибок в раздельные файлы

• Выполнить произвольную команду с ограничением использования процессорного времени 300 секунд и выводом результатов и сообщений об ошибках в один файл.

Для выполнения операции с ограничением по времени работы, можно использовать команду timeout.

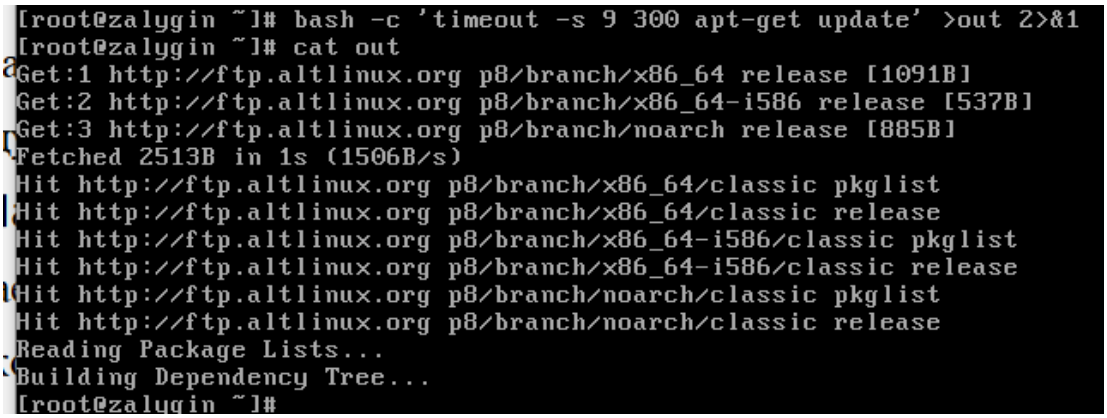

Рисунок 12 – Выполнение операции с ограничением по времени работы

Создаем задание, пишущее в файл текущую дату ежедневно.

• Настроить cron на выполнение команды ежедневно в заданное время. Продемонстрировать правильность работы.

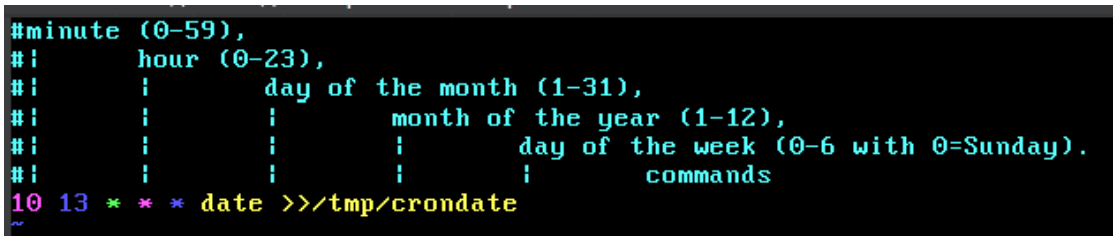

Рисунок 13 – Установка команды с запуском ежедневно в 13:10

После смотрим на содержимое файла.

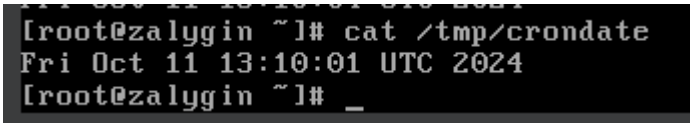

Рисунок 14 – Результат выполнения задания

ЗАКЛЮЧЕНИЕ

В результате выполнения работы были получены теоретические и практические сведения об управлении процессами, потоками и оперативной памятью в UNIX-подобных системах и в Linux в частности.