

Архитектура компьютера

Отчёт по лабораторной работе №10

Ибрахим Алькамаль

Содержание

1	Цель работы	5
2	Задание	6
3	Теоретическое введение	7
4	Выполнение лабораторной работы	9
5	Выводы	15
	Список литературы	16

Список иллюстраций

4.1	1.Создание папки и файлов	9
4.2	2.Текст файла	10
4.3	3.Создание и работа файла	10
4.4	4.Изменение прав, попытка выполнить файл	11
4.5	5.Изменение прав, выполнение файла	11
4.6	6.readme1.txt	12
4.7	7.readme2.txt	12
4.8	Исплнение файла	12
4.9	Проверка наличия файла	13

Список таблиц

1 Цель работы

Приобретение навыков написания программ для работы с файлами.

2 Задание

1. Создайте каталог для программ лабораторной работы № 10, перейдите в него и создайте файлы lab10-1.asm, readme-1.txt и readme-2.txt: `mkdir ~/work/arch-pc/lab09 cd ~/work/arch-pc/lab09 touch lab10-1.asm readme-1.txt readme-2.txt`
2. Введите в файл lab10-1.asm текст программы из листинга 10.1 (Программа записи в файл сообщения). Создайте исполняемый файл и проверьте его работу.
3. С помощью команды `chmod` измените права доступа к исполняемому файлу lab10-1, запретив его выполнение. Попробуйте выполнить файл. Объясните результат.
4. С помощью команды `chmod` измените права доступа к файлу lab10-1.asm с исходным текстом программы, добавив права на исполнение. Попробуйте выполнить его и объясните результат.
5. В соответствии с вариантом в таблице 10.4 предоставить права доступа к файлу readme1.txt представленные в символьном виде, а для файла readme-2.txt – в двоичном виде. Проверить правильность выполнения с помощью команды `ls -l`.

3 Теоретическое введение

ОС GNU/Linux является многопользовательской операционной системой. И для обеспечения защиты данных одного пользователя от действий других пользователей существуют специальные механизмы разграничения доступа к файлам. Кроме ограничения доступа, данный механизм позволяет разрешить другим пользователям доступ данным для совместной работы. Права доступа определяют набор действий (чтение, запись, выполнение), разрешённых для выполнения пользователям системы над файлами. Для каждого файла пользователь может входить в одну из трех групп: владелец, член группы владельца, все остальные. Для каждой из этих групп может быть установлен свой набор прав доступа. Владелец файла является его создатель. Для предоставления прав доступа другому пользователю или другой группе командой `chown [ключи] [:новая_группа]` или `chgrp [ключи] < новая_группа >` Набор прав доступа задается тройками битов и состоит из прав на чтение, запись и исполнение файла. В символьном представлении он имеет вид строк `gwx`, где вместо любого символа может стоять дефис. Всего возможно 8 комбинаций, приведенных в таблице 10.1. Буква означает наличие права (установлен в единицу второй бит триады `г` — чтение, первый бит `w` — запись, нулевой бит `х` — исполнение), а дефис означает отсутствие права (нулевое значение соответствующего бита). Также права доступа могут быть представлены как восьмеричное число. Так, права доступа `rw-` (чтение и запись, без исполнения) понимаются как три двоичные цифры `110` или как восьмеричная цифра `6`. Таблица 10.1. Двоичный, буквенный и восьмеричный способ записи триады прав доступа

Двоичный	Буквенный	Восьмеричный
111	<code>gwx</code>	7
110	<code>rw-</code>	6

101 r-x 5 100 r- 4 011 -wx 3 010 -w- 2 001 -x 1 000 — 0 Полная строка прав доступа в символьном представлении имеет вид: Так, например, права `rw-r-x-r-x` выглядят как двоичное число `111 101 001`, или восьмеричное `751`. Свойства (атрибуты) файлов и каталогов можно вывести на терминал с помощью команды `ls` с ключом `-l`. Так например, чтобы узнать права доступа к файлу `README` можно узнать с помощью следующей команды: `$ls -l /home/debugger/README -rwxr-xr-`

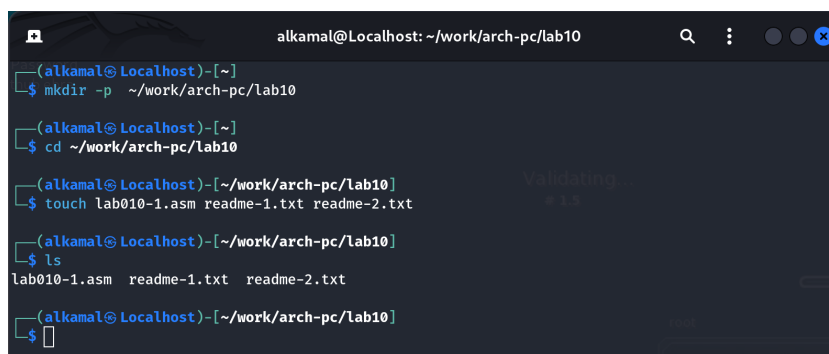
`1 debugger users 0 Feb 14 19:08 /home/debugger/README` В первой колонке показаны текущие права доступа, далее указан владелец файла и группа: Тип файла определяется первой позицией, это может быть: каталог — `d`, обычный файл — дефис (`-`) или символьная ссылка на другой файл — `l`. Следующие 3 набора по 3 символа определяют конкретные права для конкретных групп: `r` — разрешено чтение файла, `w` — разрешена запись в файл; `x` — разрешено исполнение файла и дефис (`-`) — право не дано. Для изменения прав доступа служит команда `chmod`, которая понимает как символьное, так и числовое указание прав. Для того чтобы назначить файлу `/home/debugger/README` права `rw-r`, то есть разрешить владельцу чтение и запись, группе только чтение, остальным пользователям — ничего: `$chmod 640 README # 110 100 000 == 640 == rw-r--`

`$ls -l README -rw-r 1 debugger users 0 Feb 14 19:08 /home/debugger/README` В символьном представлении есть возможность явно указывать какой группе какие права необходимо добавить, отнять или присвоить. Например, чтобы добавить право на исполнение файла `README` группе и всем остальным: `$chmod go+x README $ls -l README -rw-r-x-x`

`1 debugger users 0 Feb 14 19:08 /home/debugger/README` Формат символьного режима: `chmod`

4 Выполнение лабораторной работы

- 1) Создаю каталог для программ лабораторной работы № 10, перехожу в него и создаю файлы lab10-1.asm, readme-1.txt и readme-2.txt:



```
alkamal@Localhost: ~/work/arch-pc/lab10
(alkamal@Localhost)~$ mkdir -p ~/work/arch-pc/lab10
(alkamal@Localhost)~$ cd ~/work/arch-pc/lab10
(alkamal@Localhost)~/work/arch-pc/lab10$ touch lab10-1.asm readme-1.txt readme-2.txt
(alkamal@Localhost)~/work/arch-pc/lab10$ ls
lab10-1.asm  readme-1.txt  readme-2.txt
(alkamal@Localhost)~/work/arch-pc/lab10$
```

Рис. 4.1: 1.Создание папки и файлов

- 2) Ввожу в файл lab10-1.asm текст программы из листинга 10.1. Создаю исполняемый файл и проверяю его работу (2-3)

```

GNU nano 7.2 /home/alkamal/work/arch-pc/lab10/lab010-1.asm
; Запись в файл строки введенной на запрос
;-----
%include 'in_out.asm'
SECTION .data
filename db 'readme.txt', 0h ; Имя файла
msg db 'Введите строку для записи в файл: ', 0h ; Сообщение
SECTION .bss
contents resb 255 ; переменная для вводимой строки
SECTION .text
global _start
_start:
; --- Печать сообщения `msg`
mov eax,msg
call sprint
; ---- Запись введенной с клавиатуры строки в `contents`
mov ecx, contents
mov edx, 255
call sread
; --- Открытие существующего файла (`sys_open`)
mov ecx, 2 ; открываем для записи (2)
mov ebx, filename
mov eax, 5
int 80h
; --- Запись дескриптора файла в `esi`
mov esi, eax
; --- Расчет длины введенной строки
mov eax, contents ; в `eax` запишется количество
call slen ; введенных байтов
; --- Записываем в файл `contents` (`sys_write`)
mov edx, eax
mov ecx, contents
mov ebx, esi
mov eax, 4
int 80h
; --- Закрываем файл (`sys_close`)
mov ebx, esi
mov eax, 6
int 80h
call quit

```

Рис. 4.2: 2.Текст файла

```

alkamal@Localhost: ~/work/arch-pc/lab10
(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ nasm -f elf lab10-1.asm
(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ld -m elf_i386 -o lab10-1 lab10-1.o
(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ./lab10-1
Введите строку для записи в файл: 1234

```

Рис. 4.3: 3.Создание и работа файла

- 3) С помощью команды `chmod` изменяю права доступа к исполняемому файлу `lab10-1`, запретив его выполнение. Пытаюсь выполнить файл

```

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod 600 lab10-1

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ./lab10-1
zsh: permission denied: ./lab10-1

```

Рис. 4.4: 4.Изменение прав, попытка выполнить файл

Не получается. Ошибка “Permissin denied” означает, что в разрешении отказано. Это произошло из-за изменения прав. Как у юзера, у меня есть право только редактировать этот файл и читать, но не исполнять

- 4) С помощью команды `chmod` изменяю права доступа к файлу `lab10-1.asm` с исходным текстом программы, добавив права на исполнение. Пытаюсь выполнить его

```

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod 700 lab10-1.asm

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ./lab10-1.asm
./lab10-1.asm: 1: Syntax error: ";" unexpected

```

Рис. 4.5: 5.Изменение прав, выполнение файла

Выполнить файл просто так невозможно, поэтому программа выдает ошибку с первой строки, но выполнение началось

- 5)В соответствии с вариантом в таблице 10.4 предоставляю права доступа к файлу `readme1.txt` представленные в символьном виде, а для файла `readme-2.txt` – в двочном виде. У меня первый вариант, поэтому выполняю его Проверяю правильность выполнения с помощью команды `ls -l` (6-7)

```
alkamal@Localhost: ~/work/arch-pc/lab10
$ ls -l readme-1.txt
-rw-r--r-- 1 alkamal alkamal 0 Dec 16 17:42 readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod a+x readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod go+w readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod g-w readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod u-r readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod n-w readme-1.txt
chmod: invalid mode: 'n-w'
Try 'chmod --help' for more information.

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod n-w readme-1.txt
chmod: invalid mode: 'n-w'
Try 'chmod --help' for more information.

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod u-w readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ls -l readme-1.txt
---xrwx 1 alkamal alkamal 0 Dec 16 17:42 readme-1.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$
```

Рис. 4.6: 6.readme1.txt

```
alkamal@Localhost: ~/work/arch-pc/lab10

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ chmod 062 readme-2.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ls -l readme-2.txt
----rw--w- 1 alkamal alkamal 0 Dec 16 17:42 readme-2.txt

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$
```

Рис. 4.7: 7.readme2.txt

#Выполнение заданий для самостоятельной работы

Пишу программу и запускаю её. Всё проверяю

```
(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ nasm -f elf lab10-2.asm

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ld -m elf_i386 -o lab10-2 lab10-2.o

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ ./lab10-2
Как вас зовут? Ибрахим Алькамаль

(alkamal@Localhost)-[~/work/arch-pc/lab10]
$ cat name.txt
Меня зовут Ибрахим Алькамаль
```

Рис. 4.8: Исполнение файла

```
(alkamal@Localhost)~/work/arch-pc/lab10
$ ls
in_out.asm  lab10-1.asm  lab10-2      lab10-2.o  readme-1.txt
lab10-1     lab10-1.o    lab10-2.asm  name.txt   readme-2.txt
```

Рис. 4.9: Проверка наличия файла

Текст файла:

```
%include 'in_out.asm'

SECTION .data
filename db 'name.txt', 0h.
msg db 'Как вас зовут? ', 0h
msg1 db 'Меня зовут ', 0h

SECTION .bss
name resb 255

SECTION .text
global _start
_start:
mov eax,msg
call sprint
mov ecx, name
mov edx, 255
call sread
mov ecx, 0777o.
mov ebx, filename
mov eax, 8
int 80h
mov esi, eax
mov eax, msg1
call slen
mov edx, eax
mov ecx, msg1
```

```
mov ebx, esi
mov eax, 4
int 80h
mov eax, name.
call slen.
mov edx, eax
mov ecx, name
mov ebx, esi
mov eax, 4
int 80h
mov ebx, esi
mov eax, 6
int 80h
call quit
```

5 Выводы

Здесь кратко описываются итоги проделанной работы.

Список литературы