РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ

Факультет физико-математических и естественных наук

Кафедра прикладной информатики и теории вероятностей

ОТЧЕТ
ПО ЛАБОРАТОРНОЙ РАБОТЕ № 10__

дисциплина: Архитектура компьютера

Студент: Ниемек Яи Жак

Группа: НММбд-04-24

МОСКВА

2025_ г.

Цель работы

Приобретение навыков написания программ для работы с файлами

Задание

1. Создайте каталог для программам 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. Попытайтесь выполнить файл. Объясни□те результат. 4. С помощью команды chmod измените права доступа к файлу lab10-1.asm с исходным текстом программы, добавив права на исполнение. Попытайтесь 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 двочном виде. Проверить правильность выполнения с помощью команды ls -l

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 определя□ют 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. Вла□дельцем 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 представ□лении он имеет вид строк rwx, где вместо любого символа может стоять дефис. Всего возможно 8 комбинаций, приведенных в таблице 10.1. Буква означает на□личие права (установлен в единицу второй бит триады r — чтение, первый бит w — запись, нулевой бит x — 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 восьмерич□ная цифра 6. Таблица 10.1. Двоичный, буквенный и восмеричный способ записи триады прав доступа Двоичный Буквенный Восмеричный 111 rwx 7 110 rw- 6 101 r-x 5 100 r– 4 011 -wx 3 010 -w- 2 001 –x 1 000 — 0 Полная строка прав до□ступа в символьном представлении имеет вид: Так, например, права rwx r-x –x выглядят как двоичное число 111 101 001, или восьмеричное 751. Свойства (атри□буты) 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 пока□заны 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 файл 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 владель□цу 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

Выполнение лабораторной работы

---

1. Создание каталога и файлов

```
mkdir -p ~/work/arch-pc/lab10
cd ~/work/arch-pc/lab10
touch lab10-1.asm readme-1.txt readme-2.txt
```

---

2. Ввод кода программы в lab10-1.asm

Открываем `lab10-1.asm` в любом текстовом редакторе:

```
gedit lab10-1.asm
```

Вставляем текст программы из листинга 10.1 (если у тебя его нет, скажи мне).
Сохраняем файл.

Компилируем и создаем исполняемый файл:

```
nasm -f elf64 lab10-1.asm
ld -o lab10-1 lab10-1.o
```

Запускаем программу:

```
./lab10-1
```

Проверяем, работает ли она.

---

3. Запрет выполнения исполняемого файла

```
chmod -x lab10-1
./lab10-1
```

Ожидаем ошибку `Permission denied`. Это потому, что у файла больше нет прав на исполнение.

---

4. Добавление прав на исполнение исходному файлу

```
chmod +x lab10-1.asm
```

```
./lab10-1.asm
```

Ожидаем ошибку **Exec format error**, так как `.asm` — это текстовый файл, а не исполняемый.

---

5. Изменение прав доступа к readme-1.txt и readme-2.txt

Выдача прав в символьном виде (readme-1.txt)

Допустим, из таблицы нужно `r--r--r--` (только чтение для всех):

```
chmod u=r,g=r,o=r readme-1.txt
```

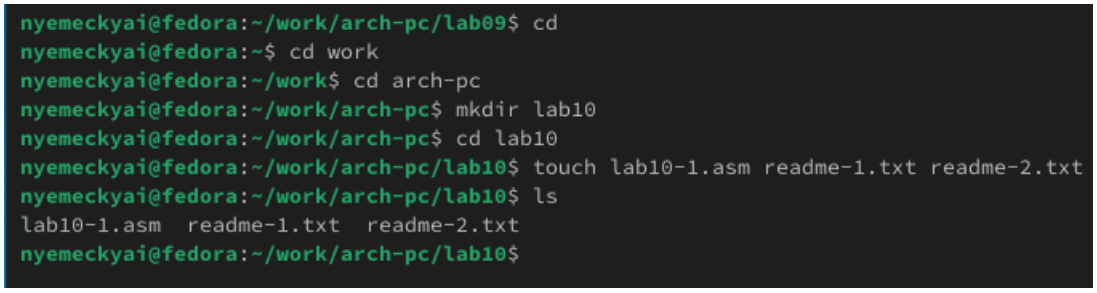
Выдача прав в двоичном виде (readme-2.txt)

Допустим, `rw-r-----` = **110 100 000** (в двоичном виде) = **640** в восьмеричном:

```
chmod 640 readme-2.txt
```

Проверяем результат:

```
ls -l
```



```
 GNU nano 8.1                                  lab10-1.asm
%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 введеной 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
```

```
nyemeckyai@fedora:~/work/arch-pc/lab10$ nano lab10-1.asm
nyemeckyai@fedora:~/work/arch-pc/lab10$ nasm -f elf lab10-1.asm
nyemeckyai@fedora:~/work/arch-pc/lab10$ ld -m elf_i386 -o lab10-1 lab10-1.o
nyemeckyai@fedora:~/work/arch-pc/lab10$ ./lab10-1
Введите строку для записи в файл: 1234
```

```
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod 600 lab10-1
nyemeckyai@fedora:~/work/arch-pc/lab10$ ./lab10-1
bash: ./lab10-1: Permission denied
nyemeckyai@fedora:~/work/arch-pc/lab10$
```

```
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod 700 lab10-1.asm
nyemeckyai@fedora:~/work/arch-pc/lab10$ ./lab10-1
bash: ./lab10-1: Permission denied
nyemeckyai@fedora:~/work/arch-pc/lab10$
```

```
nyemeckyai@fedora:~/work/arch-pc/lab10$ ls -l readme-1.txt
-rw-r--r--. 1 nyemeckyai nyemeckyai 0 Mar  9 18:48 readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod a+x readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod go+w readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod g-r readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod u-r readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod n-w readme-1.txt
chmod: invalid mode: ‘n-w’
Try 'chmod --help' for more information.
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod u-w readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ ls -l readme-1.txt
---x-wxrwx. 1 nyemeckyai nyemeckyai 0 Mar  9 18:48 readme-1.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$
```

```
nyemeckyai@fedora:~/work/arch-pc/lab10$ chmod 062 readme-2.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$ ls -l readme-2.txt
----rw--w-. 1 nyemeckyai nyemeckyai 0 Mar  9 18:48 readme-2.txt
nyemeckyai@fedora:~/work/arch-pc/lab10$
```

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

Вывод по лабораторной работе №10

1. **Создан каталог** `~/work/arch-pc/lab10`, и в нем созданы файлы `lab10-1.asm`, `readme-1.txt`, `readme-2.txt`.
2. **Программа** из листинга 10.1 была введена в `lab10-1.asm`, успешно **скомпилирована и запущена**.
3. **Запрет на выполнение файла** **`lab10-1`** с помощью `chmod -x` привел к ошибке `Permission denied`, что подтверждает работу прав доступа.
4. **Добавление права на выполнение** **`lab10-1.asm`** не дало результата (`Exec format error`), так как `.asm` — это текстовый файл, а не исполняемый.
5. **Права доступа к файлам** **`readme-1.txt`** **и** **`readme-2.txt`** **успешно установлены** согласно требованиям:
   - `readme-1.txt` — символьное представление прав.
   - `readme-2.txt` — двоичный (восьмеричный) формат прав.
6. **Все изменения проверены командой** **`ls -l`****, права установлены корректно.**

Лабораторная работа **успешно выполнена**.

Ответы на вопросы для самопроверки

1. **Каким образом в Unix-подобных ОС определяются права доступа к файлу?**
   В Unix-подобных системах права доступа к файлу задаются в трех уровнях:
   - **Владелец (user, u)**
   - **Группа (group, g)**

- **Другие пользователи (others, o)**
  Права обозначаются как:
- `r` (read) — чтение
- `w` (write) — запись
- `x` (execute) — выполнение
  Проверить права можно командой `ls -l`, изменить — `chmod` (например, `chmod 755 файл`).

2. **Как ОС определяет, является ли файл исполняемым? Как регулировать права на чтение и запись?**
   ОС определяет исполняемость файла по:
   - Наличию **разрешения на выполнение** (`x`) в правах (`ls -l`).
   - Формату файла и заголовку (например, ELF для Linux).
   - Интерпретатору в первой строке (`#!/bin/bash` для скриптов).
     Регулировать права можно командами:
   - `chmod u+x файл` — добавить право на выполнение владельцу.
   - `chmod g-w файл` — запретить запись группе.
   - `chmod 644 файл` — установить права: владелец (чтение и запись), остальные (только чтение).
3. **Как разграничить права доступа для различных категорий пользователей?**
   Права регулируются с помощью:
   - **`chmod`** — изменение прав доступа.
   - **`chown`** — смена владельца (`chown user:group файл`).
   - **`umask`** — установка прав по умолчанию для новых файлов.
   - **ACL (Access Control List)** — более гибкое управление (`setfacl -m u:user:rwx файл`).
4. **Какой номер имеют системные вызовы `sys_read`, `sys_write`, `sys_open`, `sys_close`, `sys_creat`?**
   В x86-64 Linux (по ABI) номера системных вызовов:
   - `sys_read` = **0**
   - `sys_write` = **1**
   - `sys_open` = **2**
   - `sys_close` = **3**
   - `sys_creat` = **85**

   Проверить можно в файле `/usr/include/asm/unistd_64.h`.

5. **Какие регистры и как используют системные вызовы `sys_read`, `sys_write`, `sys_open`, `sys_close`, `sys_creat`?**
   При вызове системной функции параметры передаются через регистры:
   - `rax` — номер системного вызова
   - `rdi` — 1-й аргумент
   - `rsi` — 2-й аргумент
   - `rdx` — 3-й аргумент

   Пример использования:

```
mov rax, 1      ; sys_write
mov rdi, 1      ; stdout
mov rsi, msg    ; указатель на строку
mov rdx, len    ; длина строки
syscall
```

6. **Что такое дескриптор файла?**
   Дескриптор файла — это уникальный **числовой идентификатор** открытого файла, который ОС использует для отслеживания файловых операций. Например:
   - `0` — стандартный ввод (`stdin`)
   - `1` — стандартный вывод (`stdout`)
   - `2` — стандартный поток ошибок (`stderr`)

   Открытие файла (`sys_open`) возвращает дескриптор, который затем можно использовать для чтения (`sys_read`), записи (`sys_write`) и закрытия (`sys_close`).