

Лабораторная работа No10

Работа с файлами средствами Nasm

Четвергова Мария Викторовна

Содержание

1	Цель работы	5
2	Теоретическое введение	6
3	Выполнение лабораторной работы	10
4	Выводы	15

Список иллюстраций

3.1	создание каталога и файла	10
3.2	текст программы из листинга	11
3.3	выполнение программы листинга	11
3.4	выполнение программы листинга	11
3.5	Попытка выполнить файл	12
3.6	изменение прав доступа к файлу	12
3.7	Попытка выполнить файл	13
3.8	изменение прав доступа к файлу	13
3.9	листинг данной программы	14
3.10	работа данной программы	14

Список таблиц

1 Цель работы

Приобретение навыков написания программ для работы с файлами. В ходе выполнения лабораторной работы №10 необходимо научиться работать с файлами и написать программы, работа

2 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 `новый_пользователь [: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 `гwx`, 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 `x` — исполнение), а дефис означает отсутствие права (нулевое значение соответствующего бита). Также права доступа могут быть представлены как восьмеричное число. Так, права доступа `гw` (чтение и запись, без исполнения) понимаются как три двоичные цифры `110` или как восьмеричная цифра `6`.

Полная строка прав доступа в символьном представлении имеет вид: Так,

например, права `rwX r-x -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:

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:

В операционной системе Linux существуют различные методы управления файлами, например, такие как создание и открытие файла, только для чтения или для чтения и записи, добавления в существующий файл, закрытия и удаления файла, предоставление прав доступа

Обработка файлов в операционной системе Linux осуществляется за счет использования определенных системных вызовов. Для корректной работы и доступа к файлу при его открытии или создании, файлу присваивается уникальный номер (16-битное целое число) – дескриптор файла. В таблице 10.3 приведены системные вызовы для обработки файлов

Общий алгоритм работы с системными вызовами в Nasm можно представить

в следующем виде: 1. Поместить номер системного вызова в регистр EAX; 2. Поместить аргументы системного вызова в регистрах EBX, ECX и EDX; 3. Вызов прерывания (int 80h); 4. Результат обычно возвращается в регистр EAX

Для открытия существующего файла служит системный вызов `sys_open`, который использует следующие аргументы: права доступа к файлу в регистре EDX, режим доступа к файлу в регистр ECX, имя файла в EBX и номер системного вызова `sys_open` (5) в EAX. Среди режимов доступа к файлам чаще всего используются: • (0) – `O_RDONLY` (открыть файл в режиме только для чтения); • (1) – `O_WRONLY` – (открыть файл в режиме только записи); • (2) – `O_RDWR` – (открыть файл в режиме чтения и записи). С другими режимами доступа можно ознакомиться в <https://man7.org/>. Системный вызов возвращает файловый дескриптор открытого файла в регистр EAX. В случае ошибки, код ошибки также будет находиться в регистре EAX

Для записи в файл служит системный вызов `sys_write`, который использует следующие аргументы: количество байтов для записи в регистре EDX, строку содержимого для записи в ECX, файловый дескриптор в EBX и номер системного вызова `sys_write` (4) в EAX. Системный вызов возвращает фактическое количество записанных байтов в регистр EAX. В случае ошибки, код ошибки также будет находиться в регистре EAX. Прежде чем записывать в файл, его необходимо создать или открыть, что позволит получить дескриптор файла.

Для чтения данных из файла служит системный вызов `sys_read`, который использует следующие аргументы: количество байтов для чтения в регистре EDX, адрес в памяти для записи прочитанных данных в ECX, файловый дескриптор в EBX и номер системного вызова `sys_read` (3) в EAX. Как и для записи, прежде чем читать из файла, его необходимо открыть, что позволит получить дескриптор файла.

Для правильного закрытия файла служит системный вызов `sys_close`, который использует один аргумент – дескриптор файла в регистре EBX. После вызова ядра происходит удаление дескриптора файла, а в случае ошибки, системный вызов

возвращает код ошибки в регистр EAX

Для изменения содержимого файла служит системный вызов `sys_lseek`, который использует следующие аргументы: исходная позиция для смещения EDI, значение смещения в байтах в ECX, файловый дескриптор в EBX и номер системного вызова `sys_lseek` (19) в EAX. Значение смещения можно задавать в байтах. Значения обозначающие исходную позиции могут быть следующими

3 Выполнение лабораторной работы

1. Создайте каталог для программ лабораторной работы No 10, перейдите в него и создайте файлы lab10-1.asm, readme-1.txt и readme-2.txt:

```
mvchetvergova@dk4n65 ~/work/arch-pc $ mkdir lab10
mvchetvergova@dk4n65 ~/work/arch-pc $ ls
lab04 lab05 lab06 lab07 lab08 lab10
mvchetvergova@dk4n65 ~/work/arch-pc $ cd ~/work/arch-pc/lab10
mvchetvergova@dk4n65 ~/work/arch-pc/lab10 $ touch lab10-1.asm
mvchetvergova@dk4n65 ~/work/arch-pc/lab10 $ touch readme.txt
mvchetvergova@dk4n65 ~/work/arch-pc/lab10 $ touch readme-1.txt
mvchetvergova@dk4n65 ~/work/arch-pc/lab10 $ touch readme-2.txt
mvchetvergova@dk4n65 ~/work/arch-pc/lab10 $
```

Рис. 3.1: создание каталога и файла

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.

```

lab10-1.asm      [----]  9 L: [ 1+39  40/ 40] *(1317/1317b) <EOF>
;-----
; Запись в файл строки введенной на запрос
;-----
%include 'in_out.asm'
SECTION .data
filename db 'readme-1.txt', 0h ; Имя файла
msg db 'Введите строку для записи в файл: ', 0h ; Сообщение/Архитектура ЭВМ
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. 3.2: текст программы из листинга

```

mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ nasm -f elf -g -l lab10-1.lst lab10-1.asm
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ld -m elf_i386 -o lab10-1 lab10-1.o
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ./lab10-1
Введите строку для записи в файл: kuyasdfjhkjhg

```

Рис. 3.3: выполнение программы листинга

```

cat: readme-1.txt: Нет такого файла или каталога
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ cat readme-1.txt
kuyasdfjhkjhg
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ mcedit lab10-1.asm

```

Рис. 3.4: выполнение программы листинга

3. С помощью команды `chmod` измените права доступа к исполняемому файлу `lab10-1`, запретив его выполнение. Попробуйте выполнить файл. Объясни-

те результат. ответ: В данном случае мы запретили доступ к исполнению файла. Команда `chmod` отвечает за изменение прав доступа, “u” - изменение прав доступа для владельца, “-” - отмена определённых прав, “x” - права выполнения файла.

```
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ chmod u-x lab10-1
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ./lab10-1
bash: ./lab10-1: Отказано в доступе
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $
```

Рис. 3.5: Попытка выполнить файл

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. ответ: В данном случае мы добавили владельцу права на исполнение файла. Команда `chmod` отвечает за изменение прав доступа, “u” - изменение прав доступа для владельца, “+” - добавление определённых прав, “x” - права выполнения файла.

```
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ chmod u+x lab10-1
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ./lab10-1
Введите строку для записи в файл: 999
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ cat readme-1.txt
999
```

Рис. 3.6: изменение прав доступа к файлу

5. В соответствии с вариантом в таблице 10.4 предоставить права доступа к файлу `readme-1.txt` представленные в символьном виде, а для файла `readme-2.txt` – в двоичном виде. Проверить правильность выполнения с помощью команды `ls -l`

```

mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ chmod 676 * readme-1.txt
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ls -l
итого 34
-rw-rwxrwx- 1 mvchetvergova studsci 3942 окт 26 15:55 in_out.asm
-rw-rwxrwx- 1 mvchetvergova studsci 9768 дек 16 11:47 lab10-1
-rw-rwxrwx- 1 mvchetvergova studsci 1317 дек 16 11:36 lab10-1.asm
-rw-rwxrwx- 1 mvchetvergova studsci 13744 дек 16 11:47 lab10-1.lst
-rw-rwxrwx- 1 mvchetvergova studsci 2544 дек 16 11:47 lab10-1.o
-rw-rwxrwx- 1 mvchetvergova studsci 14 дек 16 11:57 readme-1.txt
-rw-rwxrwx- 1 mvchetvergova studsci 0 ноя 25 15:34 readme-2.txt

```

Рис. 3.7: Попытка выполнить файл

```

mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ chmod 676 readme-2.txt # 101 111 111
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ls -l
итого 34
---x-----x 1 mvchetvergova studsci 3942 окт 26 15:55 in_out.asm
---x-----x 1 mvchetvergova studsci 9768 дек 16 11:47 lab10-1
---x-----x 1 mvchetvergova studsci 1317 дек 16 11:36 lab10-1.asm
---x-----x 1 mvchetvergova studsci 13744 дек 16 11:47 lab10-1.lst
---x-----x 1 mvchetvergova studsci 2544 дек 16 11:47 lab10-1.o
---x-----x 1 mvchetvergova studsci 14 дек 16 11:57 readme-1.txt
-rw-rwxrwx- 1 mvchetvergova studsci 0 ноя 25 15:34 readme-2.txt
mvchetvergova@dk2n26 ~/work/arch-pc/lab10 $

```

Рис. 3.8: изменение прав доступа к файлу

##Задание для самостоятельной работы

1. Напишите программу работающую по следующему алгоритму • Вывод приглашения “Как Вас зовут?” • ввести с клавиатуры свои фамилию и имя • создать файл с именем name.txt • записать в файл сообщение “Меня зовут” • дописать в файл строку введенную с клавиатуры • закрыть файл

```

lab10: bash — Konsole
Новая вкладка Разделить окно Копировать Вставить Найти
name.asm [----] 8 L: [ 16+32 48/ 58] *(1513/1663b) 0110 0x06E [*][X]
call sprint
; ---- Запись введенной с клавиатуры строки в 'contents'
mov eax, contents
mov ebx, 255
call sread

mov eax, 0777o ; Создание файла.
mov ebx, filename ; В случае успешного создания файла,
mov ecx, 8 ; В регистр ecx запишется дескриптор файла
int 80h

; ---- Открытие существующего файла ('sys.open')
mov ecx, 2 ; открываем для записи (2)
mov ebx, filename
mov eax, 5
int 80h
; ---- Запись дескриптора файла в 'ecx'
mov ecx, eax
; ---- Расчет длины введенной строки
mov ebx, msg2 ; в 'ebx' запишется количество
call slen ; введенных байтов

; ---- Записываем в файл 'msg2'
mov ebx, eax
mov ecx, msg2
mov ebx, esi
mov eax, 4
int 80h

; ---- Записываем в файл 'contents' ('sys.write')
mov ebx, contents
call slen
mov ebx, eax
mov ecx, contents
mov ebx, esi
mov eax, 4
int 80h
; открываем файл ('sys.close')
mov ebx, esi
mov eax, 6
int 80h
call quit
1Помощь 2Сохранить 3Блок 4Замена 5Копия 6Перенести 7Поиск 8Удалить 9МенюМС 10Выход

```

Рис. 3.9: листинг данной программы

Создать исполняемый файл и проверить его работу. Проверить наличие файла и его содержимое с помощью команд `ls` и `cat`

```

vchetvergova@dk2n26 ~/work/arch-pc/lab10 $
vchetvergova@dk2n26 ~/work/arch-pc/lab10 $ nasm -f elf -g -l name.lst name.asm
vchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ld -m elf_i386 -o name name.o
vchetvergova@dk2n26 ~/work/arch-pc/lab10 $ ./name
как вас зовут? Четвергова Мария
vchetvergova@dk2n26 ~/work/arch-pc/lab10 $ cat name.txt
меня зовут Четвергова Мария

```

Рис. 3.10: работа данной программы

4 Выводы

в ходе выполнения лабораторной работы №10 мы приобрели навыки написания программ для работы с файлами и контроля доступа к файлам.