МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

**Липецкий государственный технический университет**

Факультет автоматизации и информатики

Лабораторная работа №2

по операционным системам

“Исследование исполняемых модулей”

Студент Станиславчук С. М.

Группа АС-21-1

Руководитель Останков А. И.

Липецк 2023 г.

**2. Цель работы, задание**

Цель работы:

Изучить содержимое исполняемых файлов.

Задание:

1. Установить среду программирования gcc

2. Разработать небольшую программу на языке c (20-30 строк). В программе должны присутствовать:

- массивы или строки с заданными исходными данными

- циклы обработки массивов с вычислением результата

- условные операторы в процессе вычисления

- вывод результатов расчётов в stdout

Темы программы не должны повторяться у разных студентов, поэтому нужно её согласовать друг с другом и/или с преподавателем.

3. Необходимо отладить программу. Т.е. добиться её корректного исполнения.

4. Получить три варианта модуля программы:

- объектный файл (результат cc -c <file.c>)

- исполняемый файл, собранный статически (результат cc -static <file.c>)

- исполняемый файл, собранный динамически (результат сс <file.c>)

В двух последних случаях целесообразно использовать ключ -o для указания имени создаваемого модуля

5. Сравнить три полученных модуля одной и той же программы. по размеру

6. Исследовать содержимое трех полученных файлов с помощью команд objdump и readelf и объяснить:

- состав сегментов, включенных в исполняемые модули, какова их роль и как они располагаются в памяти

- состав программных секций, размещённых в сегментах и откуда они появились в исполняемых модулях

- таблицу символов

7. Идентифицировать секцию, содержащую код программы и попытаться его дизассемблировать. Интерпретировать полученный результат

8. Объяснить разницу в размерах трех модулей.

Отчёт должен содержать:

- исходный текст программы

- её дизассемблированный вариант

- таблицы сегментов (для двух исполняемых модулей)

- таблицы секций (в трёх вариантах)

Код программы:

#include <stdio.h>

const int arraySize = 11;

const int residual = 8;

float power(float value, int power){

float result = 1;

while (power > 0){

result \*= value;

power -= 1;

}

return result;

}

int main(){

int array[11] = {1, 2, 4, 8, 16, 17, 25, 46, 14, 5, 17};

int result = 0;

float poweredValue = power(array[arraySize-1], arraySize);

for (int i = 0; i < arraySize; i++)

{

float poweredValue = power(array[i], i);

if (result % residual == 0)

{

printf("result that == division without residual == 0 = %d", result);

printf("\n");

}

result = poweredValue;

}

return 0;

}

Дизассемблированный вариант:

Дизассемблированная секция кода командой  
**objdump -d --section=.text script.o**:

0000000000000000 <power>:

0: f3 0f 1e fa endbr64

4: 55 push %rbp

5: 48 89 e5 mov %rsp,%rbp

8: f3 0f 11 45 ec movss %xmm0,-0x14(%rbp)

d: 89 7d e8 mov %edi,-0x18(%rbp)

10: f3 0f 10 05 00 00 00 movss 0x0(%rip),%xmm0 # 18 <power+0x18>

17: 00

18: f3 0f 11 45 fc movss %xmm0,-0x4(%rbp)

1d: eb 13 jmp 32 <power+0x32>

1f: f3 0f 10 45 fc movss -0x4(%rbp),%xmm0

24: f3 0f 59 45 ec mulss -0x14(%rbp),%xmm0

29: f3 0f 11 45 fc movss %xmm0,-0x4(%rbp)

2e: 83 6d e8 01 subl $0x1,-0x18(%rbp)

32: 83 7d e8 00 cmpl $0x0,-0x18(%rbp)

36: 7f e7 jg 1f <power+0x1f>

38: f3 0f 10 45 fc movss -0x4(%rbp),%xmm0

3d: 5d pop %rbp

3e: c3 ret

000000000000003f <main>:

3f: f3 0f 1e fa endbr64

43: 55 push %rbp

44: 48 89 e5 mov %rsp,%rbp

47: 48 83 ec 50 sub $0x50,%rsp

4b: 64 48 8b 04 25 28 00 mov %fs:0x28,%rax

52: 00 00

54: 48 89 45 f8 mov %rax,-0x8(%rbp)

58: 31 c0 xor %eax,%eax

5a: c7 45 c0 01 00 00 00 movl $0x1,-0x40(%rbp)

61: c7 45 c4 02 00 00 00 movl $0x2,-0x3c(%rbp)

68: c7 45 c8 04 00 00 00 movl $0x4,-0x38(%rbp)

6f: c7 45 cc 08 00 00 00 movl $0x8,-0x34(%rbp)

76: c7 45 d0 10 00 00 00 movl $0x10,-0x30(%rbp)

7d: c7 45 d4 11 00 00 00 movl $0x11,-0x2c(%rbp)

84: c7 45 d8 19 00 00 00 movl $0x19,-0x28(%rbp)

8b: c7 45 dc 2e 00 00 00 movl $0x2e,-0x24(%rbp)

92: c7 45 e0 0e 00 00 00 movl $0xe,-0x20(%rbp)

99: c7 45 e4 05 00 00 00 movl $0x5,-0x1c(%rbp)

a0: c7 45 e8 11 00 00 00 movl $0x11,-0x18(%rbp)

a7: c7 45 b0 01 00 00 00 movl $0x1,-0x50(%rbp)

ae: ba 0b 00 00 00 mov $0xb,%edx

b3: b8 0b 00 00 00 mov $0xb,%eax

b8: 83 e8 01 sub $0x1,%eax

bb: 48 98 cltq

bd: 8b 44 85 c0 mov -0x40(%rbp,%rax,4),%eax

c1: 66 0f ef c9 pxor %xmm1,%xmm1

c5: f3 0f 2a c8 cvtsi2ss %eax,%xmm1

c9: 66 0f 7e c8 movd %xmm1,%eax

cd: 89 d7 mov %edx,%edi

cf: 66 0f 6e c0 movd %eax,%xmm0

d3: e8 00 00 00 00 call d8 <main+0x99>

d8: 66 0f 7e c0 movd %xmm0,%eax

dc: 89 45 b8 mov %eax,-0x48(%rbp)

df: c7 45 b4 00 00 00 00 movl $0x0,-0x4c(%rbp)

e6: eb 6e jmp 156 <main+0x117>

e8: 8b 45 b4 mov -0x4c(%rbp),%eax

eb: 48 98 cltq

ed: 8b 44 85 c0 mov -0x40(%rbp,%rax,4),%eax

f1: 66 0f ef d2 pxor %xmm2,%xmm2

f5: f3 0f 2a d0 cvtsi2ss %eax,%xmm2

f9: 66 0f 7e d0 movd %xmm2,%eax

fd: 8b 55 b4 mov -0x4c(%rbp),%edx

100: 89 d7 mov %edx,%edi

102: 66 0f 6e c0 movd %eax,%xmm0

106: e8 00 00 00 00 call 10b <main+0xcc>

10b: 66 0f 7e c0 movd %xmm0,%eax

10f: 89 45 bc mov %eax,-0x44(%rbp)

112: b9 08 00 00 00 mov $0x8,%ecx

117: 8b 45 b0 mov -0x50(%rbp),%eax

11a: 99 cltd

11b: f7 f9 idiv %ecx

11d: 89 d0 mov %edx,%eax

11f: 85 c0 test %eax,%eax

121: 75 23 jne 146 <main+0x107>

123: 8b 45 b0 mov -0x50(%rbp),%eax

126: 89 c6 mov %eax,%esi

128: 48 8d 05 00 00 00 00 lea 0x0(%rip),%rax # 12f <main+0xf0>

12f: 48 89 c7 mov %rax,%rdi

132: b8 00 00 00 00 mov $0x0,%eax

137: e8 00 00 00 00 call 13c <main+0xfd>

13c: bf 0a 00 00 00 mov $0xa,%edi

141: e8 00 00 00 00 call 146 <main+0x107>

146: f3 0f 10 45 bc movss -0x44(%rbp),%xmm0

14b: f3 0f 2c c0 cvttss2si %xmm0,%eax

14f: 89 45 b0 mov %eax,-0x50(%rbp)

152: 83 45 b4 01 addl $0x1,-0x4c(%rbp)

156: b8 0b 00 00 00 mov $0xb,%eax

15b: 39 45 b4 cmp %eax,-0x4c(%rbp)

15e: 7c 88 jl e8 <main+0xa9>

160: b8 00 00 00 00 mov $0x0,%eax

165: 48 8b 55 f8 mov -0x8(%rbp),%rdx

169: 64 48 2b 14 25 28 00 sub %fs:0x28,%rdx

170: 00 00

172: 74 05 je 179 <main+0x13a>

174: e8 00 00 00 00 call 179 <main+0x13a>

179: c9 leave

17a: c3 ret

Таблицы сегментов:

Для отображения таблицы сегментов используется команда   
**readelf -l script\_name**

Для отображения таблицы программных секций используется команда   
**readelf -S script\_name**

Для отображения таблицы символов используется команда   
**readelf -s script\_name**

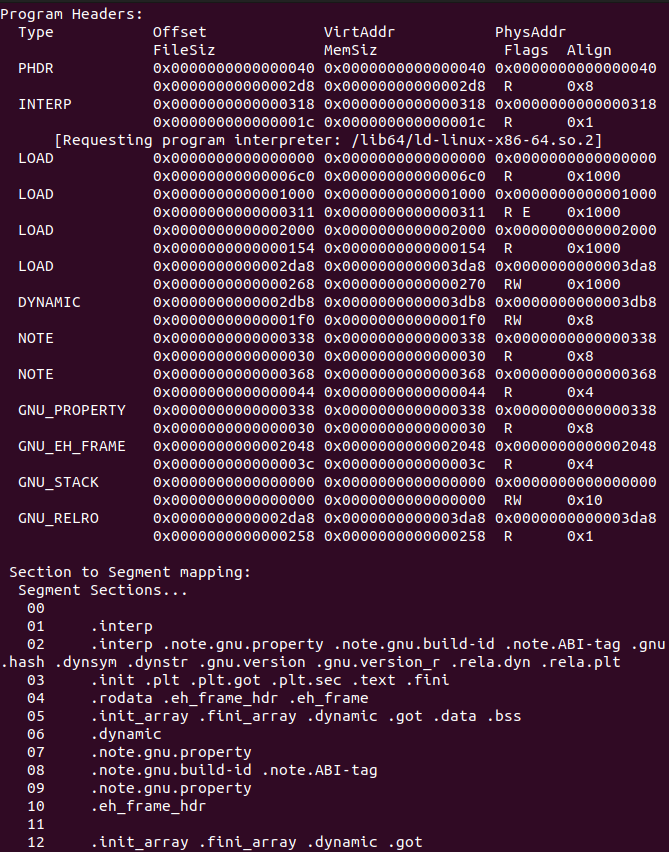


Рисунок 1 – таблица сегментов **script\_dynamic**

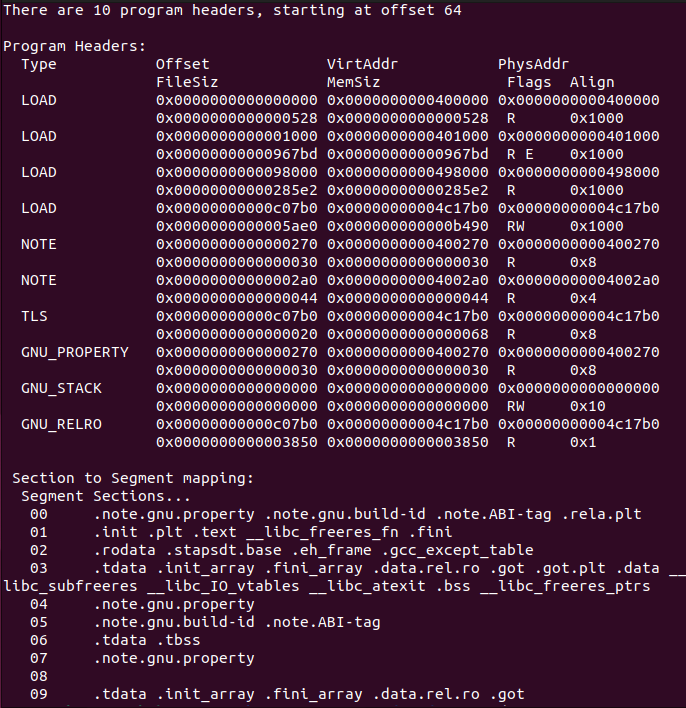
****

Рисунок 2 – таблица сегментов **script\_static**

Таблицы программных секций:

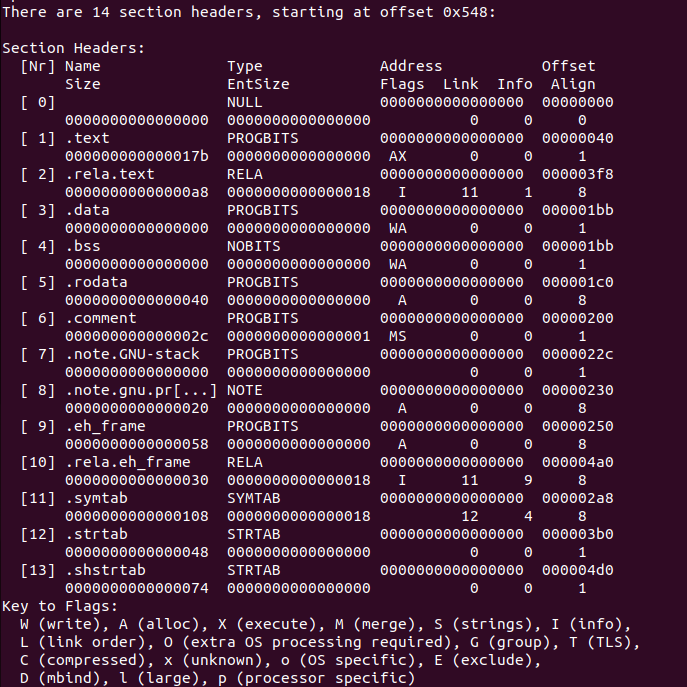
****

Рисунок 3 - Таблица программных секций **script.o**

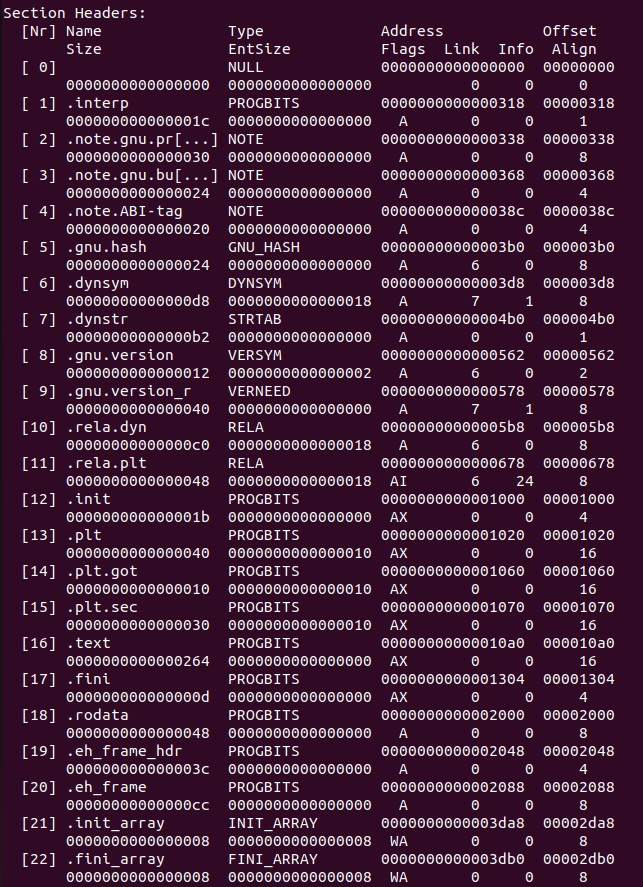
****

Рисунок 4 – Таблица программных секций **script\_dynamic** (0 – 22)

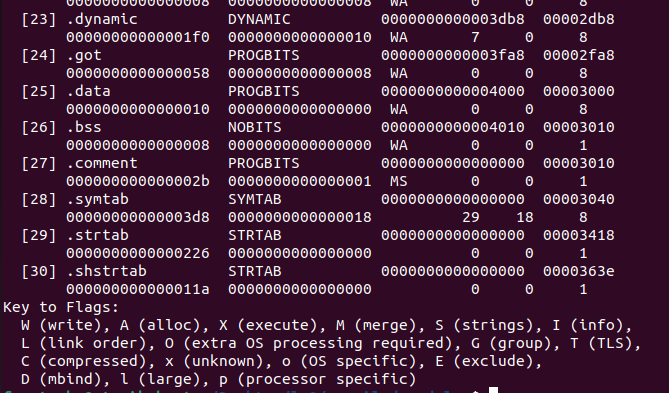
****

Рисунок 5 – Таблица программных секций **script\_dynamic** (23 – 30)

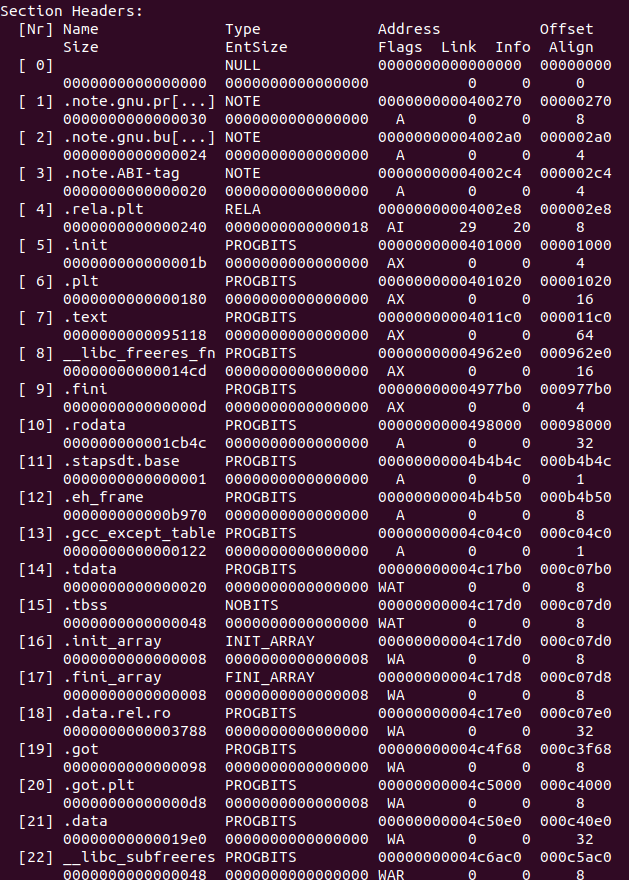


Рисунок 6 – Таблица программных секций script\_static (0 – 22)

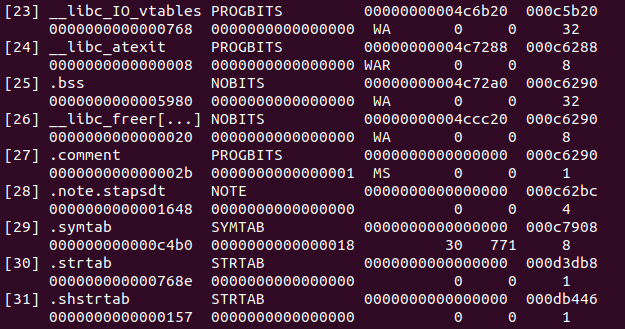


Рисунок 7 – Таблица программных секций script\_static (23 – 31)

Вывод: научился работать с терминалом Ubuntu.