

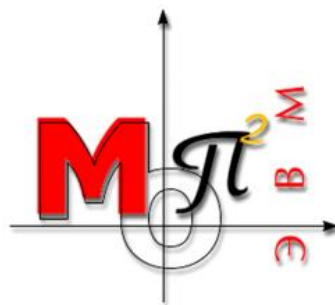
МИНОБРНАУКИ РОССИИ

Федеральное государственное автономное образовательное учреждения высшего
образования

«ЮЖНЫЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт компьютерных технологий и информационной безопасности

Кафедра математического обеспечения и применения ЭВМ



ЛАБОРАТОРНАЯ РАБОТА № 7

по дисциплине

«Безопасность информационных технологий»

на тему:

«Переполнение буфера»

Выполнил:

Студент группы

КТбо2-8

Нестеренко П. А.

Проверил:

доцент кафедры

ИБТКС

Петров Д. А.

Оценка

« ____ » _____ 2020 г.

Цель работы:

Исследование уязвимости «переполнение буфера» на примере тестовой программы и операционной системы Ubuntu/Linux.

Порядок выполнения:

1. Скорректировать тестовый файл «main.c» в соответствии с вариантом задания.
2. Скомпилировать программу командой «gcc -O0 -mpreferred-stack-boundary=2 -g -m32 -fnostack-protector main_var?.c»
3. Протестировать полученную программу путём ввода неправильного пароля, правильного пароля и неправильного пароля, длина которого превышает размер выделенного буфера.
4. Запустить отладчик gdb, задать точку останова на функцию main() и получить результат дизассемблирования функции main() и buff_overflow_test().
5. При помощи пошаговой отладки программы в gdb посмотреть содержимое стека для буфера buff_var для случаев произвольного короткого пароля и пароля, длина которого превышает размер буфера на 1 байт. Привести результат выполнения программы при вводе этих паролей.
6. В качестве отчёта по лабораторной работе привести результаты выполнения указанных выше пунктов, в том числе скорректированный исходный код и результаты работы отладчика gdb.

Индивидуальный вариант задания:

Название исходного файла	main_var2
Название буфера	buff_var2
Размер буфера	6
Правильный пароль	pass2

Скорректированный исходный код программы:

```
#include <stdio.h>
#include <string.h>

int main(void)
{
    buff_overflow_test();
    return 0;
}

int buff_overflow_test()
{
    char buff_var2[6];
    int pass = 0;

    printf("\n Enter the password : \n");
    gets(buff_var0);

    if(strcmp(buff_var2, "pass2"))
    {
        printf ("\n Wrong Password \n");
    }
    else
    {
        printf ("\n Correct Password \n");
        pass = 1;
    }

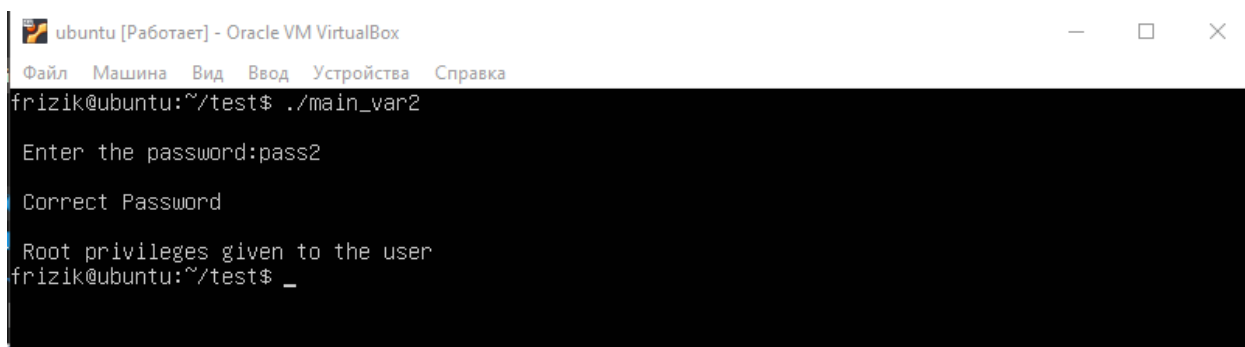
    if(pass)
    {
        /* Now Give root or admin rights to user*/
        printf ("\n Root privileges given to the user \n");
    }
}
```



The screenshot shows a terminal window titled "ubuntu [Работает] - Oracle VM VirtualBox". The terminal output is as follows:

```
frizik@ubuntu:~/test$ ./main_var2
Enter the password:qwert
Wrong Password
frizik@ubuntu:~/test$ dwa_
```

Рисунок 1 – Попытка ввода неправильного пароля, длиной менее 6 символов



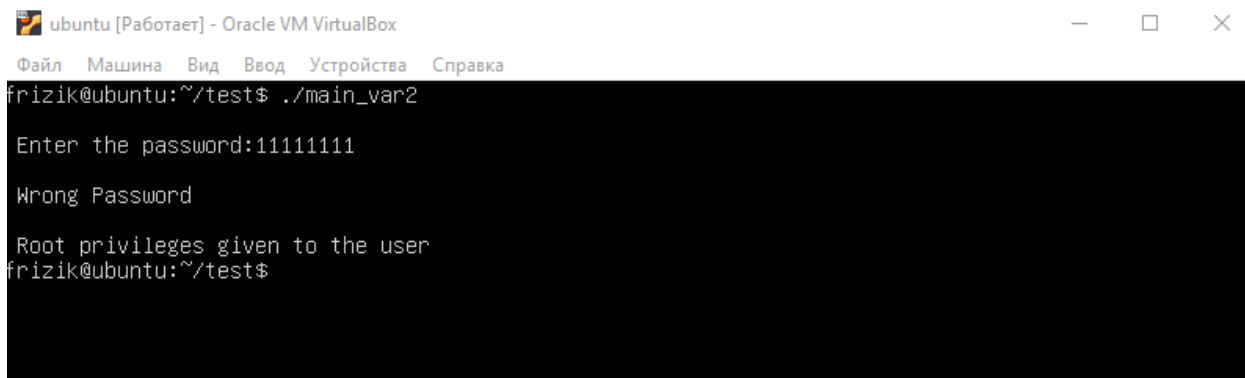
```
ubuntu [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
frizik@ubuntu:~/test$ ./main_var2

Enter the password:pass2

Correct Password

Root privileges given to the user
frizik@ubuntu:~/test$ _
```

Рисунок 2 – Ввод правильного пароля



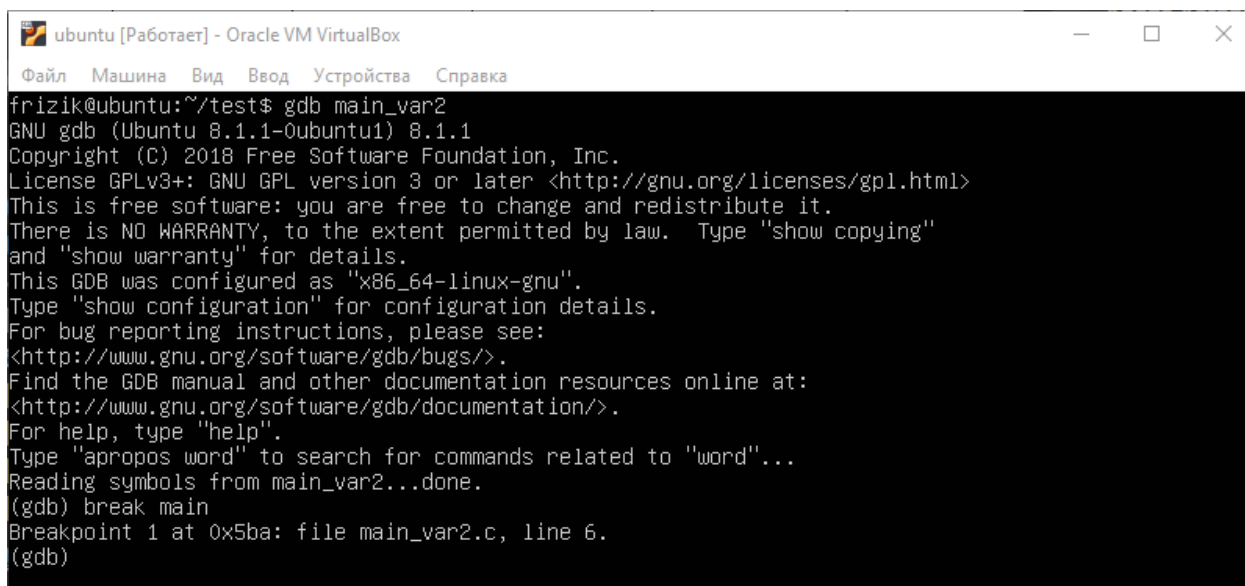
```
ubuntu [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
frizik@ubuntu:~/test$ ./main_var2

Enter the password:11111111

Wrong Password

Root privileges given to the user
frizik@ubuntu:~/test$
```

Рисунок 3 – Ввод неправильного пароля с выдачей прав



```
ubuntu [Работает] - Oracle VM VirtualBox
Файл  Машина  Вид  Ввод  Устройства  Справка
frizik@ubuntu:~/test$ gdb main_var2
GNU gdb (Ubuntu 8.1.1-0ubuntu1) 8.1.1
Copyright (C) 2018 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.  Type "show copying"
and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.
For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from main_var2...done.
(gdb) break main
Breakpoint 1 at 0x5ba: file main_var2.c, line 6.
(gdb)
```

*Рисунок 4 – Запуск **gdb** и добавление точки остановки программы*

Результат команд disassemble main и disassemble buff_overflow_test

```
(gdb) break main
Breakpoint 1 at 0x5ba: file main_var2.c, line 6.
(gdb) disassemble main
Undefined command: "disassemble". Try "help".
(gdb) disassemble main
Dump of assembler code for function main:
0x000005ad <+0>: push    %ebp
0x000005ae <+1>: mov     %esp,%ebp
0x000005b0 <+3>: call   0x653 <__x86.get_pc_thunk.ax>
0x000005b5 <+8>: add     $0x1a17,%eax
0x000005ba <+13>: call   0x5c6 <buff_overflow_test>
0x000005bf <+18>: mov     $0x0,%eax
0x000005c4 <+23>: pop     %ebp
0x000005c5 <+24>: ret
End of assembler dump.
(gdb) _

Dump of assembler code for function buff_overflow_test:
0x000005c6 <+0>: push    %ebp
0x000005c7 <+1>: mov     %esp,%ebp
0x000005c9 <+3>: push    %ebx
0x000005ca <+4>: sub     $0xc,%esp
0x000005cd <+7>: call   0x4b0 <__x86.get_pc_thunk.bx>
0x000005d2 <+12>: add     $0x19fa,%ebx
0x000005d8 <+18>: movl    $0x0,-0x8(%ebp)
0x000005df <+25>: lea     -0x18ec(%ebx),%eax
0x000005e5 <+31>: push    %eax
0x000005e6 <+32>: call   0x420 <printf@plt>
0x000005eb <+37>: add     $0x4,%esp
0x000005ee <+40>: lea     -0xe(%ebp),%eax
0x000005f1 <+43>: push    %eax
0x000005f2 <+44>: call   0x430 <gets@plt>
0x000005f7 <+49>: add     $0x4,%esp
0x000005fa <+52>: lea     -0x18d6(%ebx),%eax
0x00000600 <+58>: push    %eax
0x00000601 <+59>: lea     -0xe(%ebp),%eax
0x00000604 <+62>: push    %eax
0x00000605 <+63>: call   0x410 <strcmp@plt>
0x0000060a <+68>: add     $0x8,%esp
0x0000060d <+71>: test    %eax,%eax
0x0000060f <+73>: je      0x622 <buff_overflow_test+92>
0x00000611 <+75>: lea     -0x18d0(%ebx),%eax
0x00000617 <+81>: push    %eax
0x00000618 <+82>: call   0x440 <puts@plt>
0x0000061d <+87>: add     $0x4,%esp
0x00000620 <+90>: jmp     0x638 <buff_overflow_test+114>
0x00000622 <+92>: lea     -0x18be(%ebx),%eax
0x00000628 <+98>: push    %eax
0x00000629 <+99>: call   0x440 <puts@plt>
0x0000062e <+104>: add     $0x4,%esp
0x00000631 <+107>: movl    $0x1,-0x8(%ebp)
0x00000638 <+114>: cmpl    $0x0,-0x8(%ebp)
0x0000063c <+118>: je      0x64d <buff_overflow_test+135>
---Type <return> to continue, or q <return> to quit---
```

Данные в регистре ESP при вводе «AAA»

```
(gdb) x/x $esp
0xffff590: 0x41415960
(gdb) x/s $esp
0xffff590: "AAA"
(gdb) x/s $esp+5
0xffff596: ""
(gdb) x/x $esp+5
0xffff596: 0x00
```

Данные в регистре ESP при вводе «AAAAAA»

```
(gdb) x/x $esp
0xffff590: 0x41415960
(gdb) x/s $esp
0xffff590: "AAAAAA"
(gdb) x/x $esp+5
0xffff596: 0x41
(gdb) x/s $esp+5
```

0xffff596: "A"
