.text:00401410 ; int \_\_cdecl main(int argc, const char \*\*argv, const char \*\*envp)

.text:00401410 public \_main

.text:00401410 \_main proc near ; CODE XREF: sub\_4011A0+8E↑p

.text:00401410

.text:00401410 Str2 = byte ptr -36h

.text:00401410 Str1 = byte ptr -18h

.text:00401410 argc = dword ptr 8

.text:00401410 argv = dword ptr 0Ch

.text:00401410 envp = dword ptr 10h

.text:00401410

.text:00401410 ; \_\_unwind {

// сохраняется предыдущее значение базового указателя ebp, новое значение ebp устанавливается равным esp (указателю стека), а затем esp выравнивается по границе 16 байт и резервируется 80 (50h) байт на стеке для локальных переменных

.text:00401410 push ebp

.text:00401411 mov ebp, esp

.text:00401413 and esp, 0FFFFFFF0h

.text:00401416 sub esp, 50h //резервируем под данные

.text:00401419 call \_\_\_main

.text:0040141E mov dword ptr [esp+38h], 73736170h // 56 ssap

.text:00401426 mov dword ptr [esp+3Ch], 64726F77h //60 drow

.text:0040142E mov dword ptr [esp+40h], 333231h //64 321

.text:00401436 mov dword ptr [esp], offset Buffer ; "welcome to my crack me"

.text:0040143D call \_puts

// esp+4Ch счётчик

.text:00401442 mov dword ptr [esp+4Ch], 1 //1 в счётчик

.text:0040144A jmp chech\_counter

.text:0040144F ; ---------------------------------------------------------------------------

.text:0040144F

.text:0040144F check\_password: ; CODE XREF: \_main+E2↓j

.text:0040144F mov dword ptr [esp], offset asc\_40605C ; "---------------------------------------"

.text:00401456 call \_puts

.text:0040145B mov dword ptr [esp], offset Format ; "enter the password:"

.text:00401462 call \_printf

.text:00401467 lea eax, [esp+1Ah] //26 смещение для получения адреса введеной строки

.text:0040146B mov [esp+4], eax

.text:0040146F mov dword ptr [esp], offset aS ; "%s"

.text:00401476 call \_scanf

// загружает в eax адрес строки Str2, которая находится на стеке по смещению 50h, и помещает этот адрес в качестве второго аргумента для функции strcmp

.text:0040147B lea eax, [esp+50h+Str2]

.text:0040147F mov [esp+4], eax ; Str2

// загружает в eax адрес строки Str1, которая также находится на стеке по смещению 50h, и помещает этот адрес в качестве первого аргумента для функции strcmp

.text:00401483 lea eax, [esp+50h+Str1]

.text:00401487 mov [esp], eax ; Str1

.text:0040148A call \_strcmp

// сохраняет результат сравнения, возвращенный strcmp, на стеке по смещению 48h

.text:0040148F mov [esp+48h], eax

.text:00401493 cmp dword ptr [esp+48h], 0

.text:00401498 jnz short error\_msg

.text:0040149A mov dword ptr [esp], offset aCongratsYouCra ; "congrats you cracked the password"

.text:004014A1 call \_puts

.text:004014A6 jmp short exit

.text:004014A8 ; ---------------------------------------------------------------------------

.text:004014A8

.text:004014A8 error\_msg: ; CODE XREF: \_main+88↑j

.text:004014A8 mov dword ptr [esp], offset aWrongPass ; "wrong pass!!"

.text:004014AF call \_puts

.text:004014B4 mov eax, 5

.text:004014B9 sub eax, [esp+4Ch] //76 вычесть счётчик

.text:004014BD mov [esp+44h], eax // 68 сохранили оставшееся количество попыток

.text:004014C1 mov eax, [esp+44h]

.text:004014C5 mov [esp+4], eax // аргумент для %s (сразу нельзя из [esp+44h])

.text:004014C9 mov dword ptr [esp], offset aYouGotDLeft ; "you got %d left\n"

.text:004014D0 call \_printf

.text:004014D5 cmp dword ptr [esp+44h], 0

.text:004014DA jnz short add\_counter

.text:004014DC mov dword ptr [esp], offset aYouAreOutOfGue ; "you are out of guesses"

.text:004014E3 call \_printf

.text:004014E8

.text:004014E8 add\_counter: ; CODE XREF: \_main+CA↑j

.text:004014E8 add dword ptr [esp+4Ch], 1

.text:004014ED

.text:004014ED chech\_counter: ; CODE XREF: \_main+3A↑j

.text:004014ED cmp dword ptr [esp+4Ch], 5 // проверка счётчика

.text:004014F2 jle check\_password

.text:004014F8

.text:004014F8 exit: ; CODE XREF: \_main+96↑j

.text:004014F8 mov eax, 0

.text:004014FD leave

.text:004014FE retn

.text:004014FE ; } // starts at 401410

.text:004014FE \_main endp