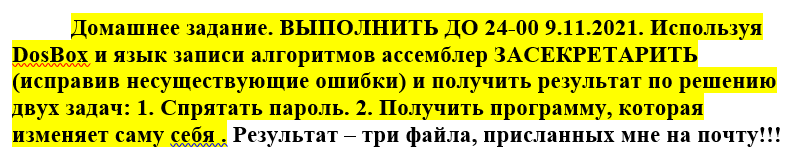
Жуковский Павел Сергеевич, 4 курс, 12 группа

Домашнее задание с лекции на 09.11.2021

Пароль от архива: **041**

Постановка задачи:



Алгоритм решения:

**Для начала решим первую задачу – «спрятать пароль».**

Для этого я (во многом вдоновившись идеями из вашей первой свиньи) написал следующий исходный текст (HIDEPSWD.ASM):

.MODEL SMALL

.STACK 400h

.DATA

TargetPswd DB 255, ?, 255 DUP (0)

PswdRepository DW 06568h, 06553h, 06868h, 06548h, 06565h, 06849h, 06868h, 06553h

DW 06565h, 06854h ; Password: S, H, I, S, T

PswdRepositorySize = $ - PswdRepository

HackReport DB "The password has been confirmed", 0Dh, 0Ah, "$"

ErrorReport DB "Entered password is incorrect, try more", 0Dh, 0Ah, "$"

.CODE

DEFINE\_SYMB:

PUSH CX

PUSH DX

XOR AL, BH

MOV CL, 80h

XOR DL, DL

.DS\_LP:

TEST AL, CL

JZ .DS\_LP\_MOVE

INC DL

.DS\_LP\_MOVE:

SHR CL, 1

TEST CL, CL

JNZ .DS\_LP

TEST DL, 1

JNZ .DS\_USE\_BL

XOR AL, AH

JMP .DS\_FINISH

.DS\_USE\_BL:

XOR AL, BL

.DS\_FINISH:

POP DX

POP CX

RET

PSWD\_MATCHING:

PUSH BX

MOV SI, DX

LEA DI, PswdRepository

MOV CX, PswdRepositorySize

SHR CX, 2

.MTCH\_LOOP:

LODSB

MOV BL, AL

PUSH BX

XCHG DI, SI

LODSW

MOV BX, AX

LODSW

CALL DEFINE\_SYMB

POP BX

XCHG DI, SI

CMP BL, AL

JNE .MTCH\_ERROR

LOOP .MTCH\_LOOP

CMP BYTE PTR [SI], 0Dh

JNE .MTCH\_ERROR

POP BX

CLC

RET

.MTCH\_ERROR:

POP BX

STC

RET

START:

MOV AX, @DATA

MOV DS, AX

MOV AH, 0Ah

LEA DX, TargetPswd

INT 21h

ADD DX, 2

CALL PSWD\_MATCHING

JC .MATCHING\_ERROR

LEA DX, HackReport

JMP .OUTPUT

.MATCHING\_ERROR:

LEA DX, ErrorReport

.OUTPUT:

MOV AH, 09h

INT 21h

MOV AX, 4C00h

INT 21h

END START

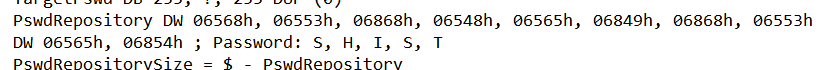
Затем (с помощью TASM.EXE и TLINK.EXE) я получил сначала объектный модуль (HIDEPSWD.OBJ), а затем исполнимый модуль (HIDEPSWD.EXE):



Для линковки мне понадобились также следующие компоненты: DPMILOAD.EXE, DPMIMEM.DLL, DPMI16BI.OVL.

В качестве пароль я загадал следующий: «**SHIST**»

В исходном тексте он хранится в следующем виде:



Настоящие байты пароля (5 байтов) скрыты среди других байтов (всего 10 байтов, включая байты пароля).

Суть в том, чтобы скрыть байты пароля среди других (ненужных байтов), а потом нужные байты переместить в регистр AL (на основе того, из чего состоит байт), и, когда все байты будут обработаны и размещены по регистрам, из него сверять настоящий пароль. Лишь байты настоящего пароля пройдут обработку таким образом, чтобы попасть в регистр AL.

Когда исполнимый модуль был получен, я загрузил программу в оперативную память компьютера и протестировал её на разных входных данных. Для начала на правильном пароле:





Затем на каком-нибудь неправильном пароле (например на тех же символах, но в нижнем регистре, а не верхнем):





Таким образом, я получил программу, которая скрывает байты настоящего пароля среди других байтов (ненужных), а во время своей работы загружает нужные байты в регистр AL и из него сверяет с тем, что вводит пользователь.

**Теперь решим вторую задачу – «получить программу, которая изменяет саму себя».**

Для этого я (опять-таки, глянув ваш исходный текст из второй свиньи) написал следующий исходный текст (SELFCHNG.ASM):

.MODEL SMALL

.STACK 400h

.DATA

SUCCESS\_CHNG DB "Program: success modifying!!!$"

.CODE

START:

MOV AX, @DATA

MOV DS, AX

XOR DX, DX

MOD\_ATTEMPT:

MOV AX, 0

TEST AX, AX

JNZ MODIFY\_DONE

TEST DX, DX

JNZ FINISH

MOV BYTE PTR MOD\_ATTEMPT, 40h

MOV WORD PTR MOD\_ATTEMPT + 1, 9090h

INC DX

JMP MOD\_ATTEMPT

MODIFY\_DONE:

MOV AH, 09h

LEA DX, SUCCESS\_CHNG

INT 21h

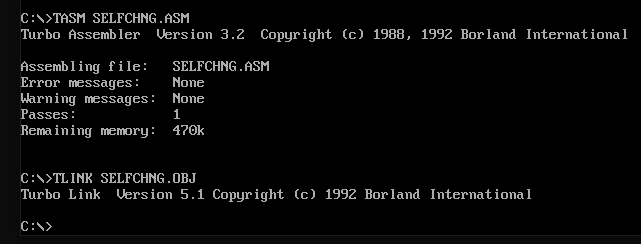
FINISH:

MOV AX, 4C00h

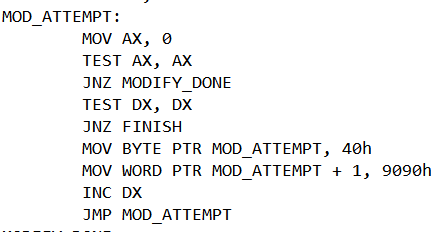
INT 21h

END START

Затем (с помощью TASM.EXE и TLINK.EXE) я получил сначала объектный модуль (SELFCHNG.OBJ), а затем исполнимый модуль (SELFCHNG.EXE):

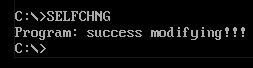


Основные изменения программы происходят здесь:



В процессе выполнения изменяется содержимое регистра AX на единицу и, после того, как программа обнаруживает, что содержимое регистра AX сменилось, она переходит на метку MODIFY\_DONE.

Когда исполнимый модуль был получен, я загрузил программу в оперативную память компьютера и протестировал её:



В письме я постараюсь прикрепить архив, который будет включать в себя весь этот отчёт, исполняемый модуль и исходный текст.

Пароль от архива: **041** (на всякий случай, укажу пароль на самом верху этого отчёта тоже)