ISSN: 2277-9655
Impact Factor: 1.852

IJESRT

INTERNATIONAL JOURNAL OF ENGINEERING SCIENCES & RESEARCH TECHNOLOGY

Code Emulation Technique For Computer Virus Detection

Ankur Singh Bist
Govind Ballabh Pant University of Agriculture and Technology, India
ankur1990bist@gmail.com

Abstract

Computer viruses are big threat to computer world. Researchers doing work in this area have made various efforts in the direction of classification and detection methods of these viruses. Graph mining, system call arrangement and graphical analysis are some latest research activities in this field. The computability theory the semi computable and computable functions are quite important in our context of analyzing malicious activity. A mathematical model like random access stored program machine with the association of attached background is used by Ferenc Leitold while explaining modeling of viruses in his paper. Computer viruses like polymorphic viruses and metamorphic viruses have more efficient techniques for their evolution so it is required to use strong models to understand their evolution and then apply detection followed by the process of removal. Code Emulation is one of the strongest ways to analyze computer viruses but the anti-emulation activities made by virus designers are also active.

Keywords: Computer virus, code emulation.

Introduction

The code emulator should have the ability to run the virus code being analysed in an emulated environment. In this way, there is a strong chance that the virus will expose itself about its functionalities. With the help of virtual flags and registers, the code emulator will execute the instruction set of the CPU. Code emulation may be a costly solution. This technique proved itself useful for the detection of complex viruses and its related forms. To implement a metamorphic virus identifier though code emulation it is required to take care about code obfuscation techniques. Code obfuscation techniques like equivalent code substitution, dead code insertion, junk block insertion and dead subroutine insertion are the primary targets of code emulator. A generic detection consists of four parts-

1. *processor emulator*
2. *memory emulator*
3. *system emulator*
4. *decision mechanism*

The purpose of code emulator is to take morphed copies of virus file closer to the base virus file in term of statistics.

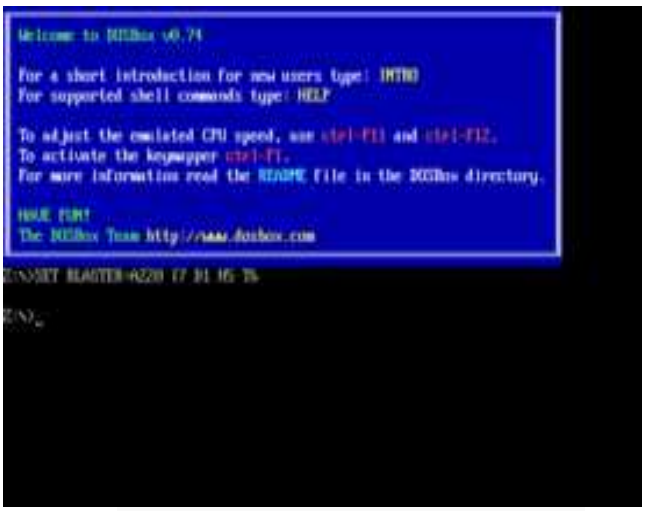

Figure-1 Emulation of command-line interface using DOSbox [1].

Role of Code Emulation in Computer Virus Detection

The main issues that are managed by code emulator----

There should be formulation of assembly level language instructions at large level at the same instance code emulator should have capacity to emulate all the essential CPU registers. The emulator should be able to classify or modify the instructions/subroutines, which are because of code obfuscation methods.

ISSN: 2277-9655
Impact Factor: 1.852

1. equivalent code substitution
2. dead code insertion
3. junk block insertion
4. dead subroutine insertion

The emulator should also preserve the basic flavour of virus program.

The Emulation of the W95/Fabi Virus [2]

```
Iteration Number                    Flags
Registers
Opcode      Instruction

Iteration: 1, IP=00405200
AX>00000000     BX>00000000     CX>00000000
DX>00000000
SI>00000000     DI>00000000     BP>0070FF87
SP>0070FE38
FC          cld

Iteration: 2, IP=00405201
AX>00000000     BX>00000000     CX>00000000
DX>00000000
SI>00000000     DI>00000000     BP>0070FF87
SP>0070FE38
E800000000  call    00405206h

Iteration: 3, IP=00405206
AX>00000000     BX>00000000     CX>00000000
DX>00000000
SI>00000000     DI>00000000     BP>0070FF87
SP>0070FE34
5D          pop     ebp

Iteration: 4, IP=00405207
AX>0000000      BX>00000000     CX>000000000
DX>00000000
SI>0000000 DI>00000000 BP>00405206 SP>0070FE38
81ED06104000  sub     ebp, 00401006h

Iteration: 5, IP=0040520D
AX>00000000     BX>00000000     CX>00000000
DX>00000000
SI>00000000     DI>00000000     BP>00004200
SP>0070FE38
8DB52A104000  lea     esi,[ebp+0040102A]

Iteration: 6, IP=00405213
AX>00000000     BX>00000000     CX>00000000
DX>00000000
SI>0040522A     DI>00000000     BP>00004200
SP>0070FE38
B95E250000  mov     ecx,255Eh
Iteration: 7, IP=00405218
AX>00000000     BX>00000000     CX>0000255E
DX>00000000
SI>0040522A     DI>00000000     BP>00004200
SP>0070FE38
BB72FD597A  mov     ebx,7A59FD72h

Iteration: 8, IP=0040521D
AX>00000000     BX>7A59FD72     CX>0000255E
DX>00000000
SI>0040522A     DI>00000000     BP>00004200
SP>0070FE38
311E        xor     [esi],ebx

Iteration: 9, IP=0040521F
AX>00000000     BX>7A59FD72     CX>0000255E
DX>00000000
SI>0040522A     DI>00000000     BP>00004200
SP>0070FE38
AD          lodsd

AX>03247C80     BX>7A59FD72     CX>0000255E
DX>00000000
SI>0040522E     DI>00000000     BP>00004200
SP>0070FE38
81C3C3D5B57B  add     ebx, 7BB5D5C3h

Iteration: 11, IP=00405226
AX>03247C80     BX>F60FD335     CX>0000255E
DX>00000000
SI>0040522E     DI>00000000     BP>00004200
SP>0070FE38     O S
E2F5        loop    0040521Dh

Iteration: 12, IP=0040521D
AX>03247C80     BX>F60FD335     CX>0000255D
DX>00000000
SI>0040522E     DI>00000000     BP>00004200
SP>0070FE38     O S
311E        xor     [esi],ebx
```

Now days there are several virtual machines used by malware researchers. Some of them are—

1. VMware
2. Qemu
3. Virtual Box
4. SandBoxes

ISSN: 2277-9655
Impact Factor: 1.852

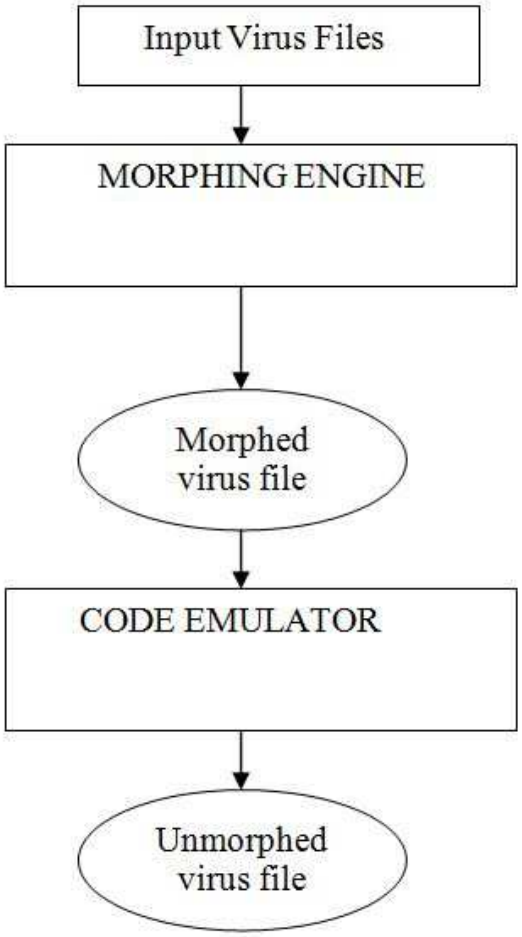

Figure-2

Virus designers are continuously analysing the virtual machines behaviour to make a counter attack to code emulation method adopted by virus designers. Some anti-emulation techniques are [3] -

1. Anti-Sandboxes
2. Anti-VMware
3. Anti-Virtual Box
4. Anti-Anubis SandBox
5. Anti-JoeBox Sandbox
6. Anti-Norman Sandbox
7. Anti-Softice

Conclusion

Code Emulation is one of the strongest techniques used for computer virus detection because this technique has the capacity to make behaviour analysis in virtual environment but the evolution of anti-emulation techniques is a big issue to be tackled to sustain the efficiency of concerned method.

References

[1] www.wikipedia.com
[2] http://computervirus.uw.hu/ch11lev1sec4.html
[3] Anoirel Issa ," Anti-virtual machines and emulations" springer 2012.