Advanced Malware and their Detection Technique

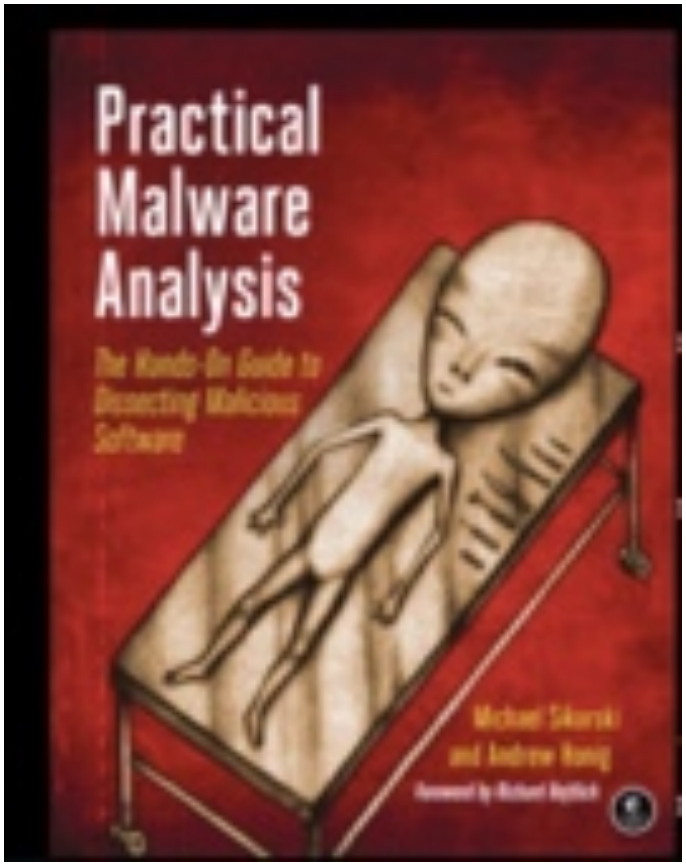
By

Dr. Ashu Sharma

C3i Center, Dept. of CSE, IIT Kanpur, India

Text Book

- Practical Malware Analysis by Michael Sikorski and Andrew Honig – ISBN 978-1593272906



Overview

- Introduction
- Basic Analysis
- Malware Static Analysis
- Malware Dynamic Analysis

Introduction: Background

- Malicious software which enters the computer system without users authorization and takes undesirable actions.

  - The development of computer security has a military origin, and since 1950 it is a major concern.
  - US government was a major force behind security research and technology.
  - In 1970 first malware was born.
  - In 1980s PC came into existence which were small enough to fit on the desk.
  - In 1990s, Internet has made a revolutionary impact on the PC users.

Why do people write malware?

- • In the 90s
  - For the fun or skill showing
    - Spread to other machines & display a message

Why do people write malware?

- Today
  - $$$

Organizations buy malware

- Steal passwords,
- credit cards,
- bank info,
- ransoms,
- intellectual property,
- trade secrets

They can use these info or sell it

Types of Malware

- 1st Generation/Static Malware:
  - Backdoor, Trojan horse, Rootkit, Scareware, Adware, Virus, Worm, etc.
- 2nd Generation/Dynamic Malware:
  - Encrypted, Oligormorphic, Polymorphic and Metamorphic Malware.

Malware Structure

Three Parts

– Attack Vector: The means by which a malware spreads, enabling it to replicate, also referred as Infection Vector.

– Trigger: The event or condition that determines when the payload is activated or delivered.

– Malicious Activities: The payload may involve damage or may involve benign but NOTICEABLE activity.

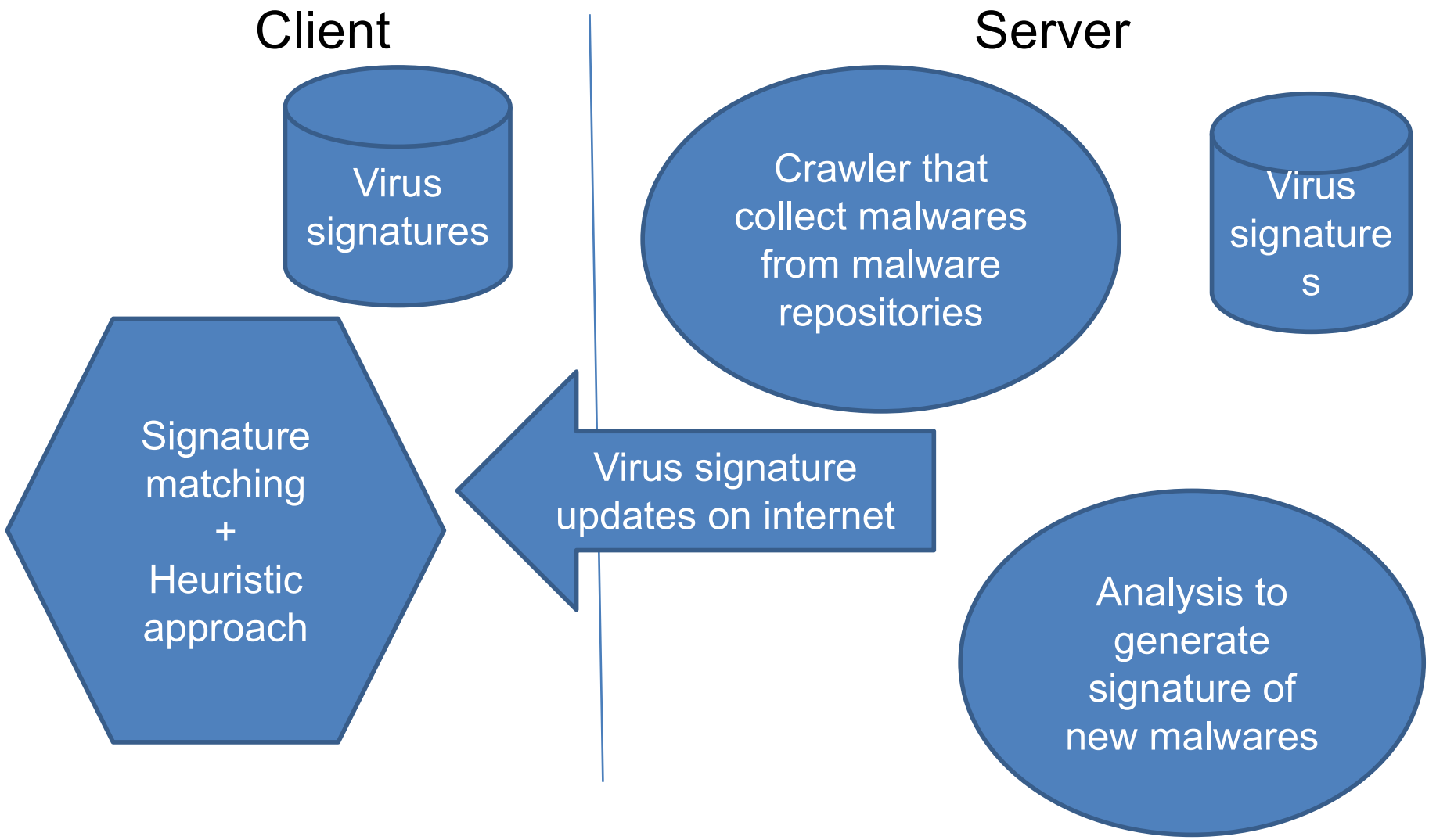
Virus Structure

```
#include <sys/types.h>                          /* standard POSIX headers */
#include <sys/stat.h>
#include <dirent.h>
#include <fcntl.h>
#include <unistd.h>
struct stat sbuf;                               /* for lstat call to see if file is sym link */

search(char *dir_name)
{                                               /* recursively search for executables */
    DIR *dirp;                                  /* pointer to an open directory stream */
    struct dirent *dp;                          /* pointer to a directory entry */

    dirp = opendir(dir_name);                   /* open this directory */
    if (dirp == NULL) return;                   /* dir could not be opened; forget it */
    while (TRUE) {
        dp = readdir(dirp);                     /* read next directory entry */
        if (dp == NULL) {                       /* NULL means we are done */
        chdir ("..");                           /* go back to parent directory */
        break;                                  /* exit loop */
    }
    if (dp->d_name[0] == '.') continue;         /* skip the . and .. directories */
    lstat(dp->d_name, &sbuf);                   /* is entry a symbolic link? */
    if (S_ISLNK(sbuf.st_mode)) continue;        /* skip symbolic links */
    if (chdir(dp->d_name) == 0) {               /* if chdir succeeds, it must be a dir */
        search(".");                            /* yes, enter and search it */
    } else {                                        /* no (file), infect it */
        if (access(dp->d_name,X_OK) == 0) /* if executable, infect it */
            infect(dp->d_name);
    }
    closedir(dirp);                             /* dir processed; close and return */
}
```

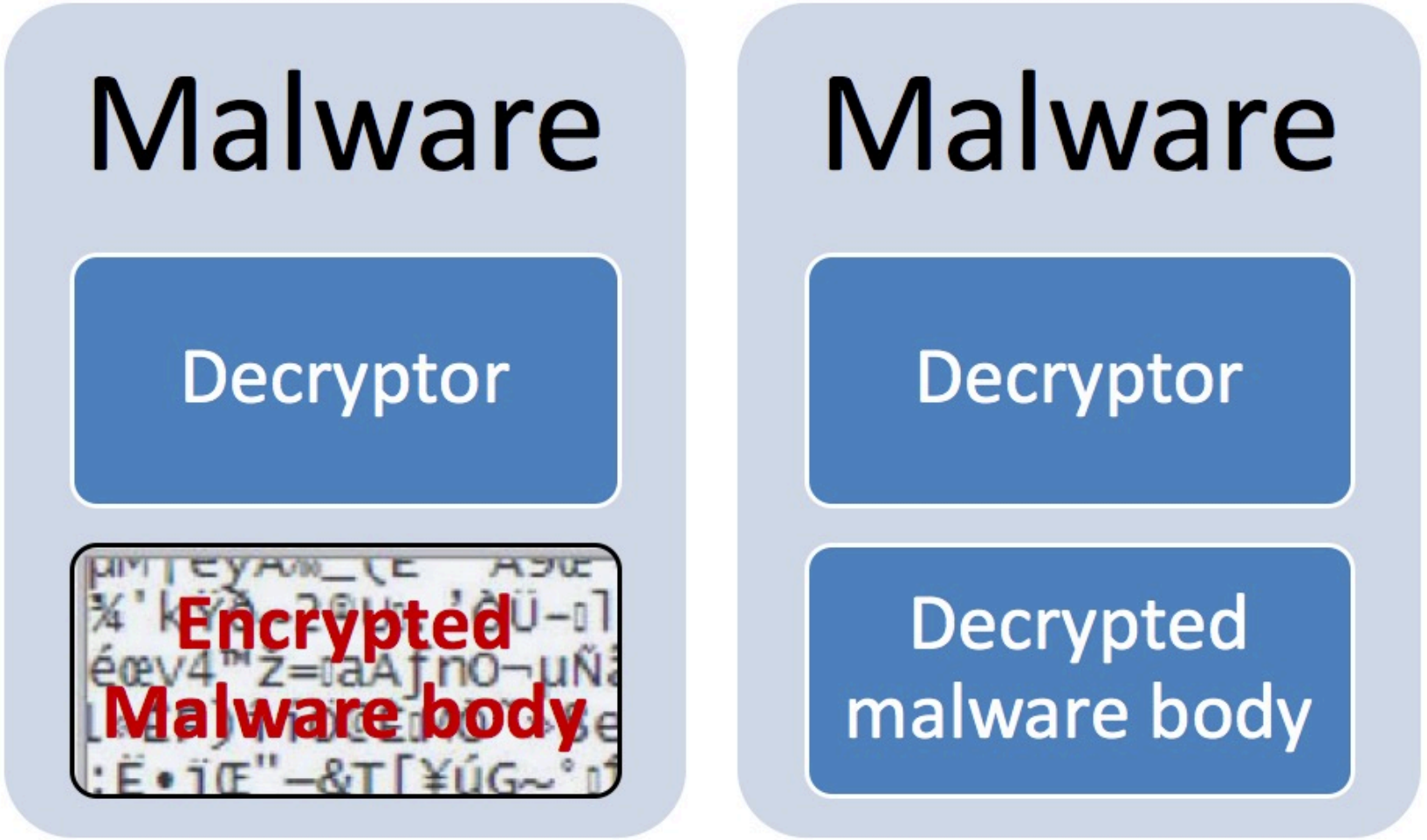
Antivirus Defense System



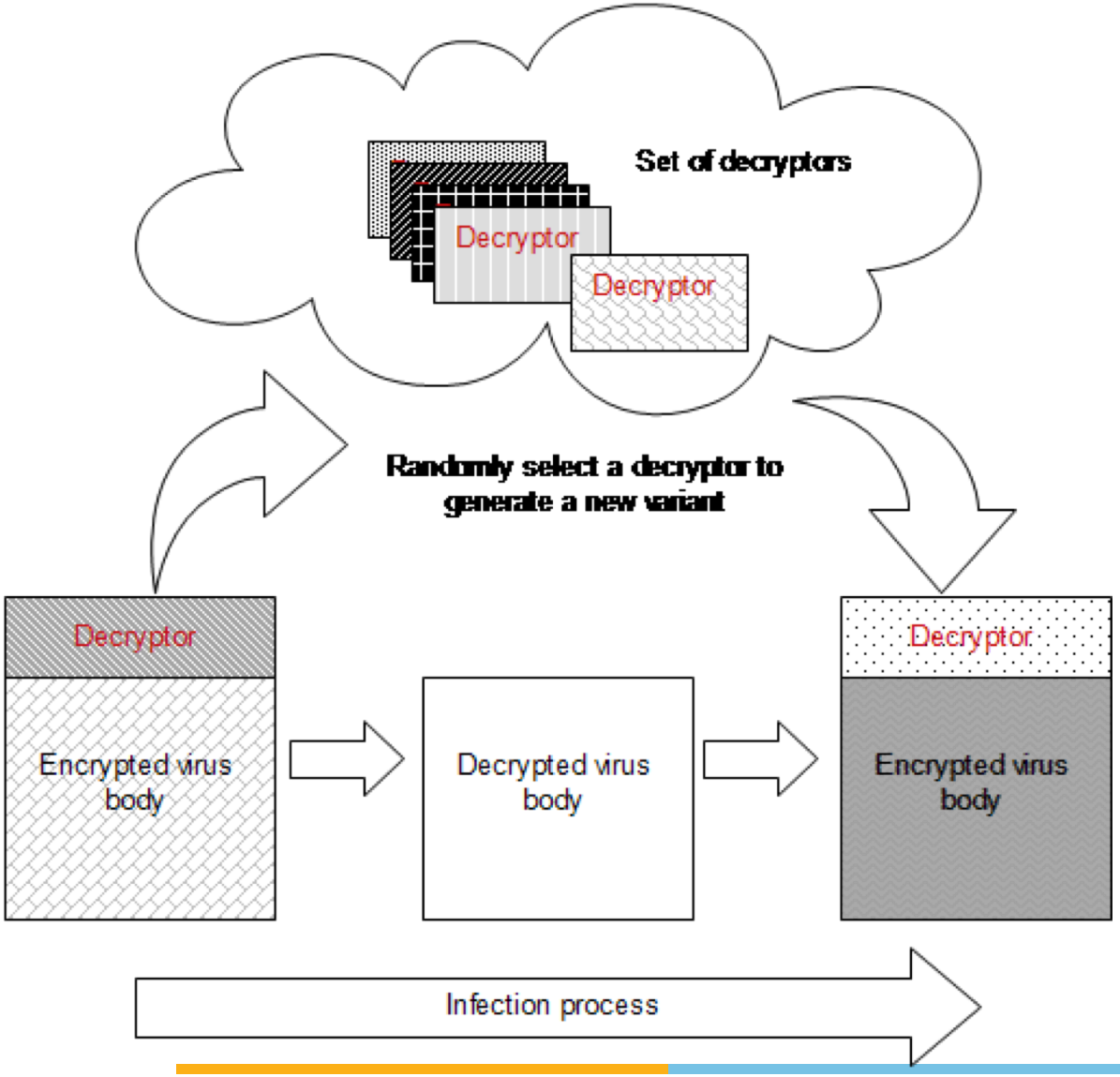
Types of Malware

- 1st Generation/Static Malware:
  - Backdoor, Trojan horse, Rootkit, Scareware, Adware, Virus, Worm, etc.
- 2nd Generation/Dynamic Malware:
  - Encrypted, Oligormorphic, Polymorphic and Metamorphic Malware.

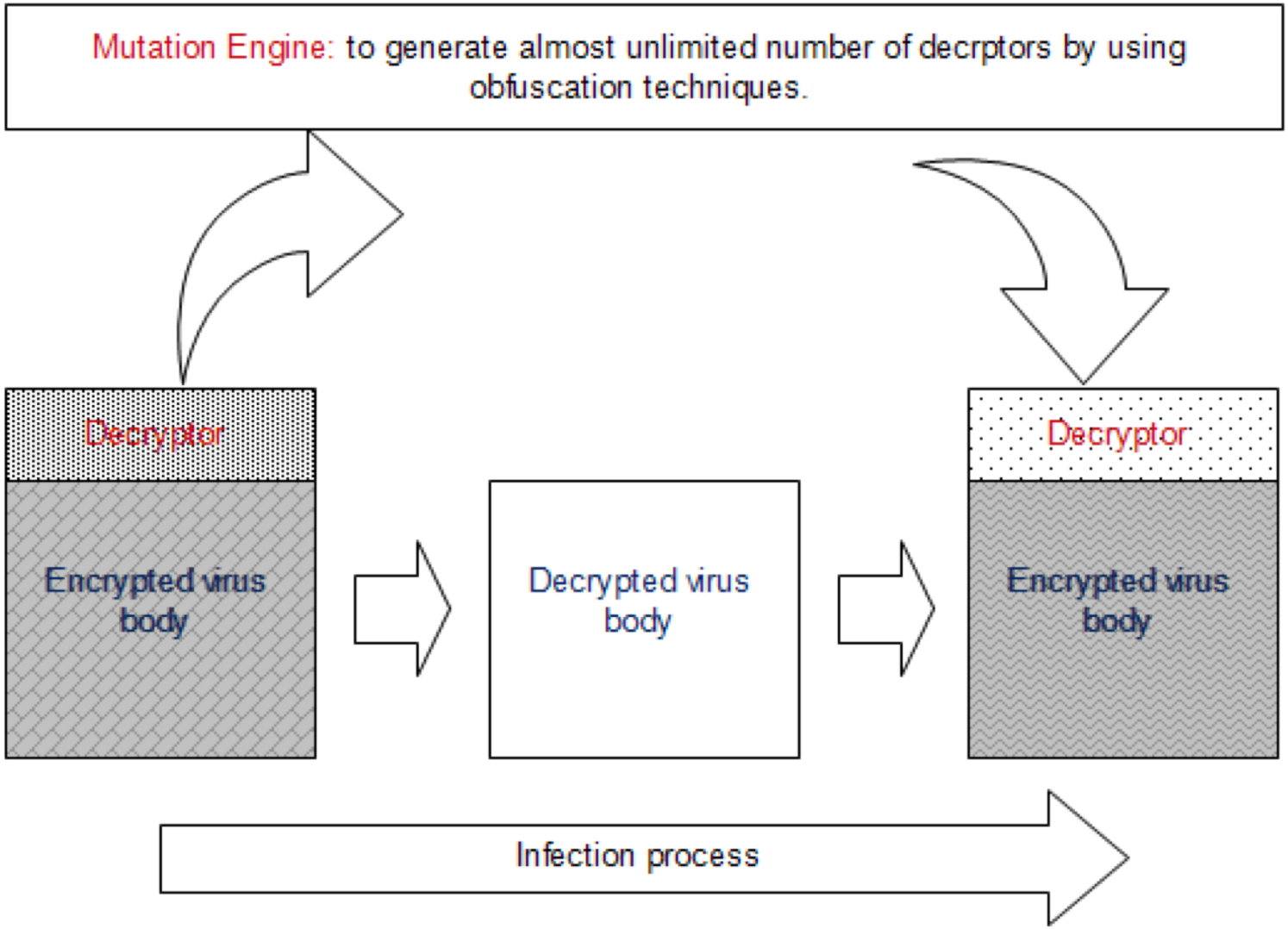
2nd Generation: Encrypted Malware



2nd Generation: Oligormorphic Malware



2nd Generation: Polymorphic Malware



Obfuscation Techniques

- Register usage exchange
- This method was used by the Win95/RegSwap virus, which was created by the virus writer Vecna and released in 1998.
- **Different generations of the virus will use the same code but with different registers**

a)

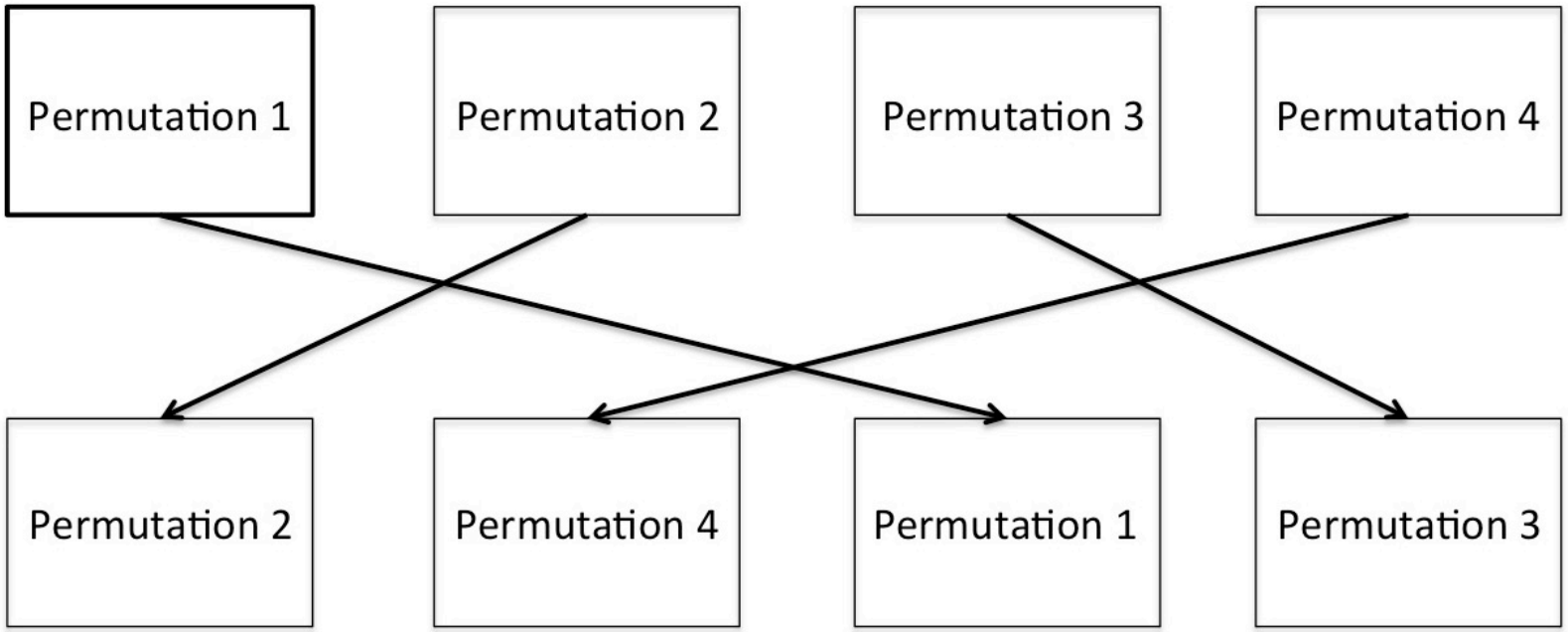
```
5A                  pop edx
BF04000000          mov edi,0004h
8BF5                mov esi,ebp
B80C000000          mov eax,000Ch
81C288000000        add edx,0088h
8B1A                mov ebx,[edx]
899C8618110000      mov [esi+eax*4+00001118],ebx
```

b)

```
58                  pop eax
BB04000000          mov ebx,0004h
8BD5                mov edx,ebp
BF0C000000          mov edi,000Ch
81C088000000        add eax,0088h
8B30                mov esi,[eax]
89B4BA18110000      mov [edx+edi*4+00001118],esi
```

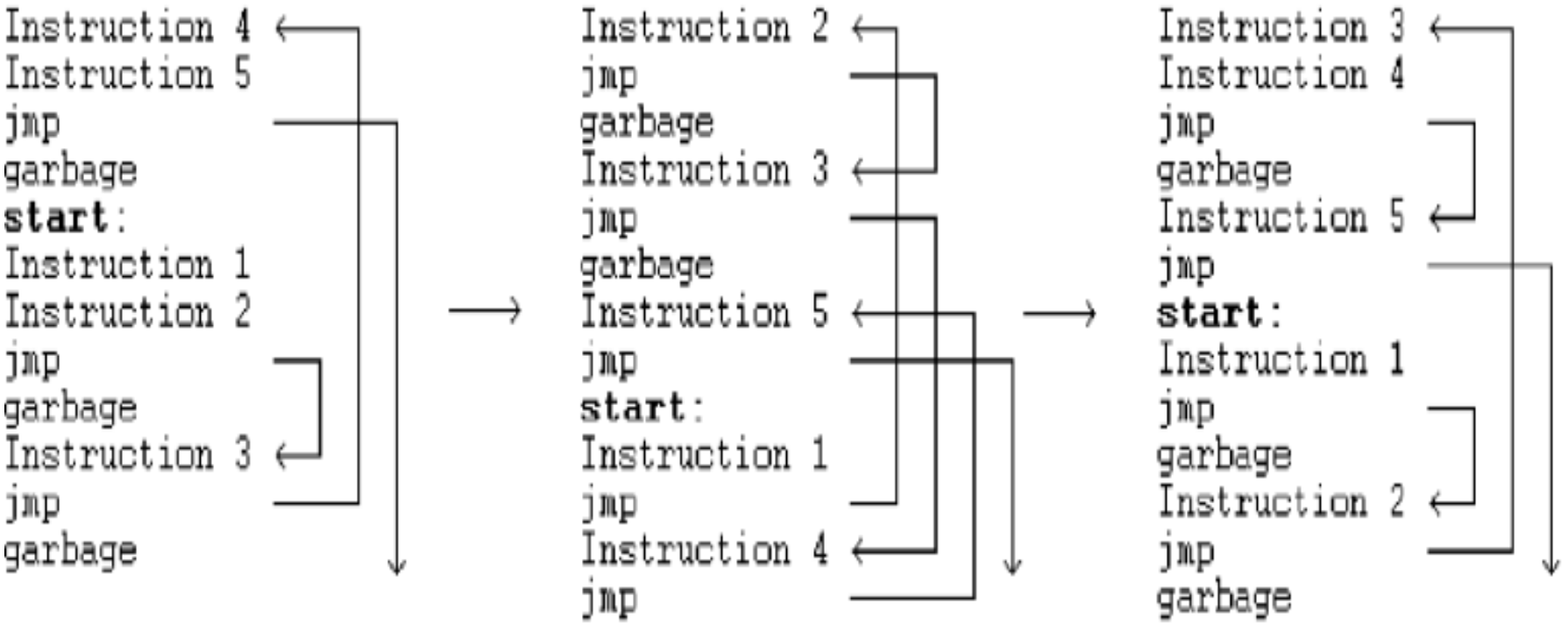
Obfuscation Techniques

- Permutation Techniques



Obfuscation Techniques

- Insertion of Jump Instructions



Obfuscation Techniques

- Subroutine Inlining and Outlining
  - In this technique, calls of the subroutines are replaced by its codes or vice versa

```
/* some instructions */
call Function 1
call Function 2
/* some instructions */
Function 1:
 mov eax, ebx
 add eax, 12h
 push eax
 ret
Function 2:
 mul ecx
 mov edx, eax
 ret
```

(a) Code variant A

```
/* some instructions */
        mov eax, ebx
        add eax, 12h
        push eax
        mul ecx
        mov edx, eax
/* some instructions */
```

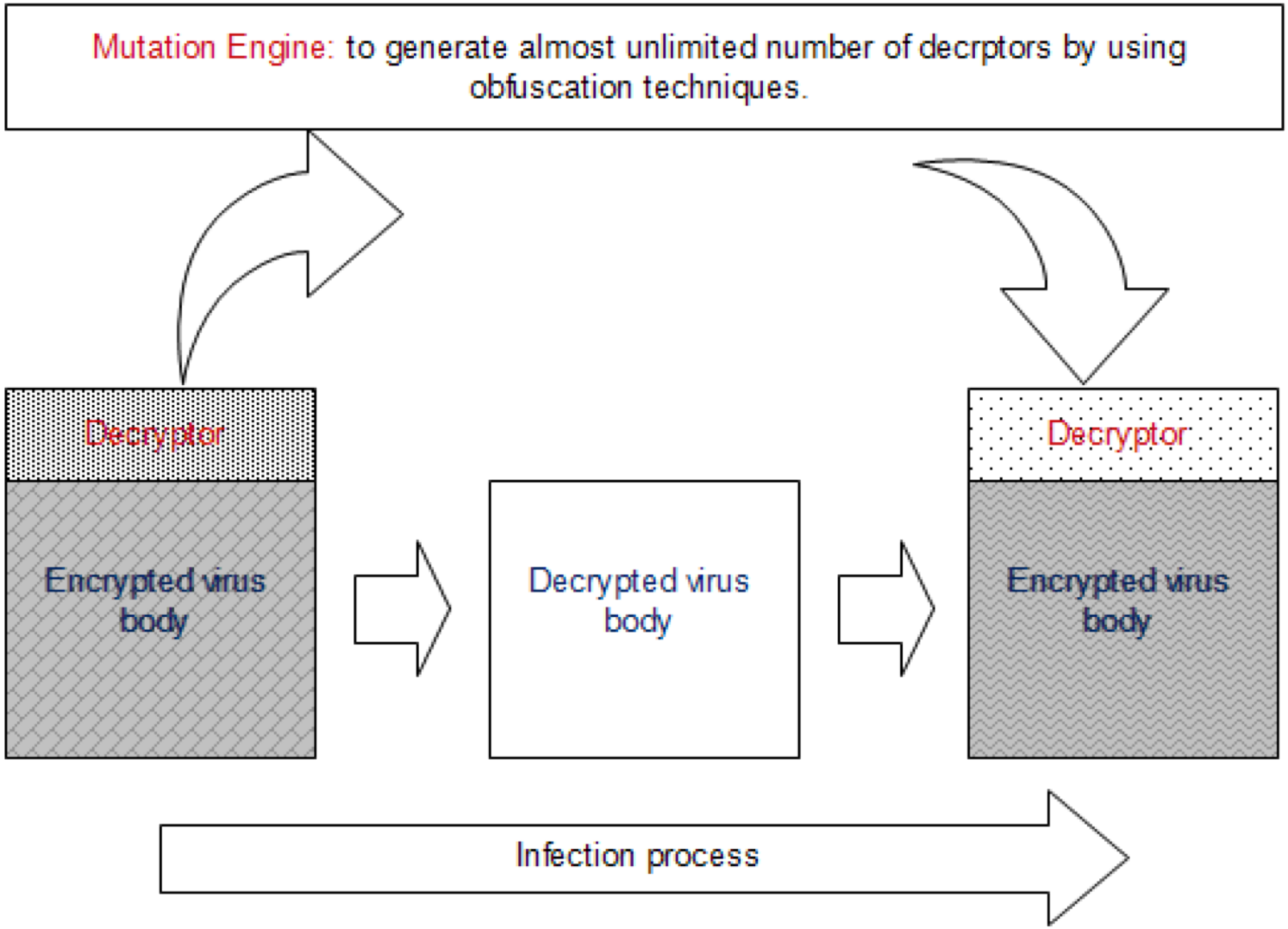
(b) Code variant B

Obfuscation Techniques

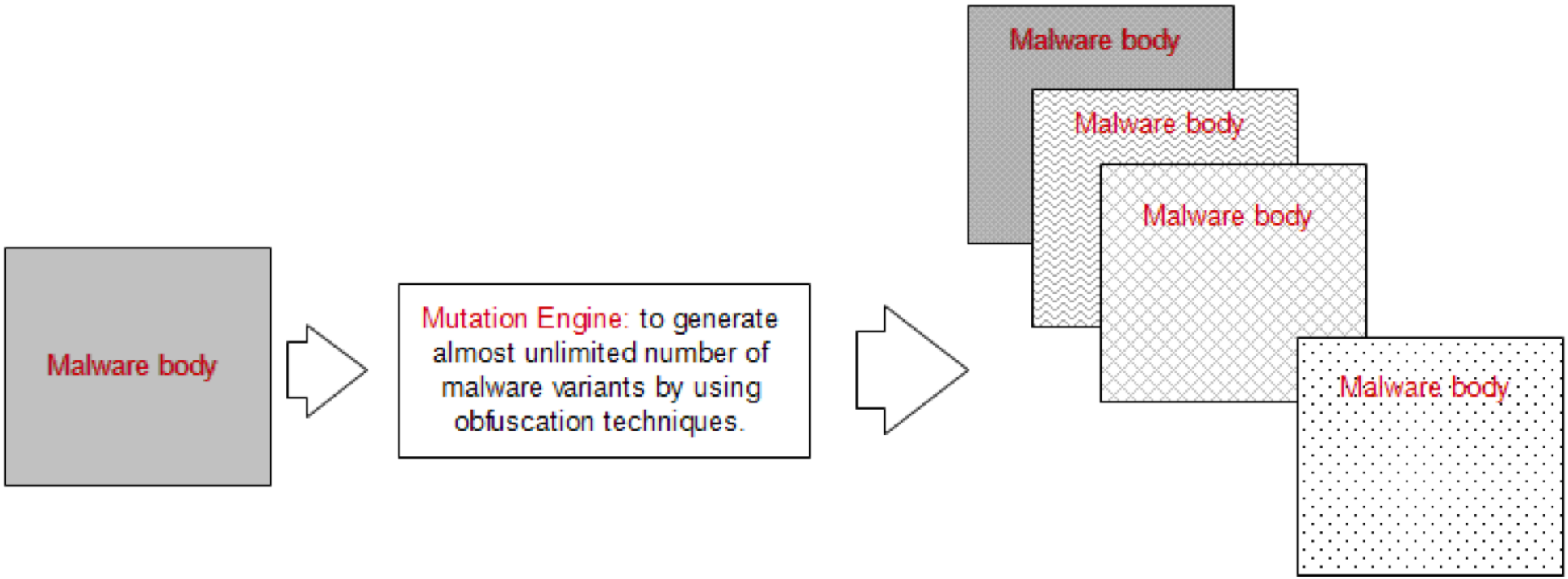
- Insertion of dead codes

```
00401005    8BF0            MOV ESI,EAX
00401007    3E:8A00         MOV AL,BYTE PTR DS:[EAX]
0040100A    84C0            TEST AL,AL
0040100C  v 74 49           JE SHORT Test.00401057
0040100E    53              PUSH EBX
0040100F    3E:8F05 74F940  POP DWORD PTR DS:[40F974]
00401016    90              NOP
00401017    D3DB            RCR EBX,CL
00401019    0FCB            BSWAP EBX
0040101B    68 59104000     PUSH Test.00401059
00401020    5B              POP EBX
00401021    3E:8903         MOV DWORD PTR DS:[EBX],EAX
00401024    90              NOP
00401025    43              INC EBX
00401026    0FBDC2          BSR EAX,EDX
00401029    A9 46A978DC     TEST EAX,DC78A946
0040102E    8BC2            MOV EAX,EDX
00401030    52              PUSH EDX
00401031    90              NOP
00401032    B6 86           MOV DH,86
00401034    B3 27           MOV BL,27
00401036    B8 7CFAA17F     MOV EAX,7FA1FA7C
0040103B  v EB 01           JMP SHORT Test.0040103E
0040103D    90              NOP
0040103E    0FBCC2          BSF EAX,EDX
00401041    3E:C705 FC8841  MOV DWORD PTR DS:[4188FC],0
0040104C    2D 210DE8B9     SUB EAX,B9E80D21
00401051    69DA E577D49D   IMUL EBX,EDX,9DD477E5
```

2nd Generation: Polymorphic Malware



2nd Generation: Metamorphic Malware



Tools to Create Metamorphic Malwares

- GenVir.
- VCL (Virus Creation Laboratory),
- PS-MPC (Phalcon-Skism Mass-Produced Code Generator),
- NGVCK (Next Generation Virus Creation Kit),
- G2 (generation second virus kit),
- VCS (Virus Construction Set)

Links :

http://83.133.184.251/virensimulation.org/vxcd5b.html?id=tidx&page=4

For Manual obfuscation: online tool

https://www.pelock.com/obfuscator/

Threat & Challenges in Malware detection

- An exponential increase in the number of malware.
- Non recoverable computer infection
- The attacks/threats are not only limited to individual level, but there are state sponsored highly skilled hackers developing customized malware e.g. Stuxnet, duqu, etc.
- Advanced Malware are capable to bypass traditional detection system.

Basic Analysis: General/Naive Procedure

- Submit the code to a virus program or online scanner such as http://www.virustotal.com; signature analysis may help determine the name and functionality of the malware

- Perform additional online research to determine the malware's purpose and capabilities

Where to Get Malware Samples for Analysis?

- https://github.com/ashubits/samples
- https://zeltser.com/malware-sample-sources/
- http://www.tekdefense.com/downloads/malware-samples/
- http://thezoo.morirt.com/

  It contains close to 100 malware binary and source codes.
- http://openmalware.org/

  You can search to find the malware binary code you want

Basic Analysis Tools

- File (Unix/Linux command) - Used to determine the file type
  - command: `file <program name>`
  - "/usr/share/file/magic.mgc" file offers approximately 5,000 different file types that Linux will recognize with the file command

- Strings (Windows & Unix) - find the printable strings in an object or other binary file
  - command: `strings -a <program name>`
    - `-a` - this option causes strings to look for strings in all sections of the object file

- BinText (Windows) - Finds ASCII, Unicode and Resource strings in a file, http://www.mcafee.com/us/downloads/free-tools/bintext.aspx

Basic Analysis

- What is the type of the file?

  file: PE32 executable (GUI) Intel 80386, for MS Windows

- What suspicious imports are used in file?
  a. ShellExecuteExA - Can be used to run applications
  b. Socket APIs - Make network connections
  c. File API - read/modify files

- What are a few strings that stick out to you and why?
  a. 60.248.52.95 - Potential network signature
  b. http://www.ueopen.com/test.html - Potential network signature
  c. cmd.exe - The malware could be trying to run shell commands
  d. *(SY)# - Potential network signature, possible used for a remote shell prompt

Basic Analysis: Conclusion

- What happens when you run this malware?

  Connects to 60.248.52.95, offers up a remote shell, then deletes itself

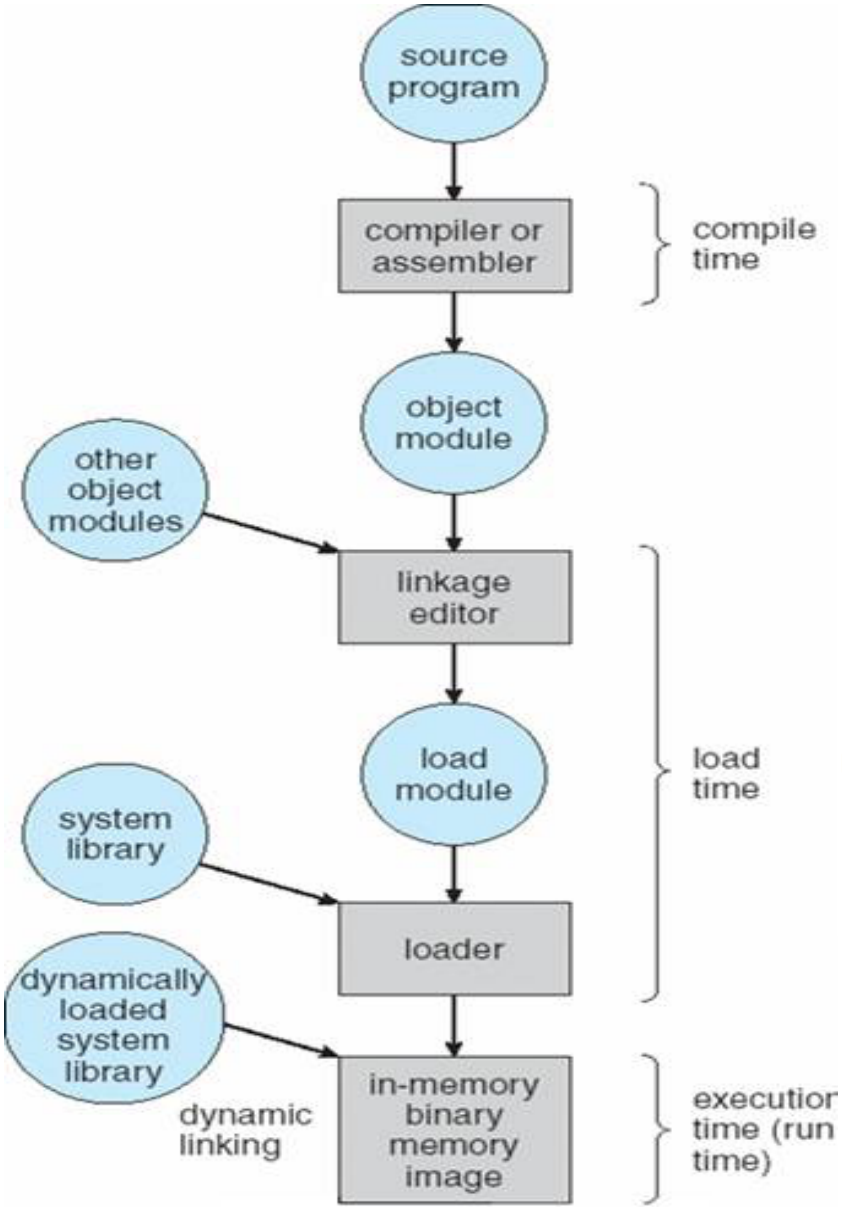
- What do you think is the purpose of this malware?
  - To act as a backdoor by offering a remote shell to the attacker

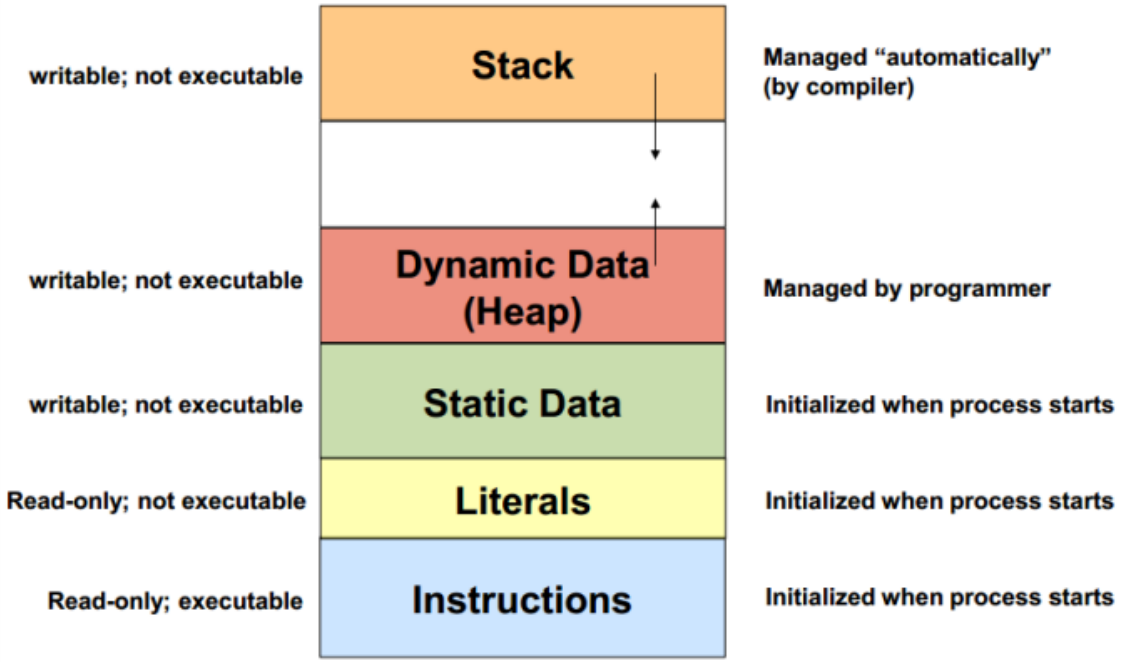
Malware Analysis

- Static Analysis:

- Dynamic Analysis

Program to Process Procedure





Static Malware Analysis

- Analysis of malware performed without actually executing the untrusted code
- Analysis can be performed on any platform because you are not intending to run the malware which may be platform specific (e. g., a Win32 executable)
- Some questions to be answered include:
  - What type of file is this (batch file, shell script, Windows executable, Linux ELF, Javascript, etc.)
  - What does it do?
  - Does it spread itself via physical media or network resources?
  - Does it steal, alter, or delete information?
- Determine the type of file you are examining, its internal structures (sections and headers) or logics using reverse engineering.

Windows Portable Executable (PE) Analysis Tools

Three PE Analysis Tools Worth Looking At

- https://blog.malwarebytes.com/threat-analysis/2014/05/five-pe-analysis-tools-worth-looking-at/
- PE Studio (free version available, portable)
  - https://www.winitor.com/
- Exeinfo PE - view detailed information about an Windows exe file including packers used
  - http://exeinfo.atwebpages.com/
- PEBrowse Professional: static-analysis and disassembler for Win32/Win64 executables
  - http://www.smidgeonsoft.prohosting.com/pebrowse-pro-file-viewer.html

Static Malware Analysis

- What Malware writers do against static analysis?

- Stripping: Removes all human-readable symbols from object code.
  - Combats reverse engineering.

- Packing with UPX, etc.
  - upx.sourceforge.net
  - Compresses source code (achieves ratios of 20% - 40%)

Reverse Engineering Tool

- Windows Binary
  - IDA PRO
  - OBJDUMP

- Android apps
  - APKTOOL

BASIC DYNAMIC ANALYSIS - PRACTICAL STEPS

- Basic dynamic analysis can assist and confirm the findings obtained from basic static analysis.

- Step 1: Baseline
  - Before executing the malware, take a first snapshot of registry using **Regshot**.

- Step 2: System status
  - During the running of malware, start **Process Monitor** and **Process Explorer**. Note any changes occurred.

BASIC DYNAMIC ANALYSIS - PRACTICAL STEPS

- Step 3: Network traffic
  - Using **Wireshark** to log network traffic generated by the malware.
- Step 4: Comparison
  - Waiting for the malware to finish making any system changes. Take a second snapshot using **Regshot**. Compare the differences between two snapshots.

Virtual Machine

- What is a virtual machine?

  –Simply, a computer in your computer

  –Really, a (usually) segregated virtual environment that emulates real hardware

- Why are we using a virtual machine?

  –Safety, reliability, consistency, it's easy

  –Keep the malware in a contained environment

  –Snapshots: Completely 100% revert the VM to an earlier state, If things go bad, no one cares

Virtual Machine

- Downloads of Windows VM Images
  - link https://developer.microsoft.com/en-us/microsoft-edge/tools/vms/
  - The password to your VM is "Passw0rd!"

Ubuntu , Kali and other linux VM images

- Link https://www.osboxes.org/virtualbox-images/

Installation Manual

- Link https://oalabs.openanalysis.net/2018/07/16/oalabs_malware_analysis_virtual_machine/

DYNAMIC ANALYSIS - PREPARATION

- Using dedicated physical or virtual machines (VMs) to analyze malware.

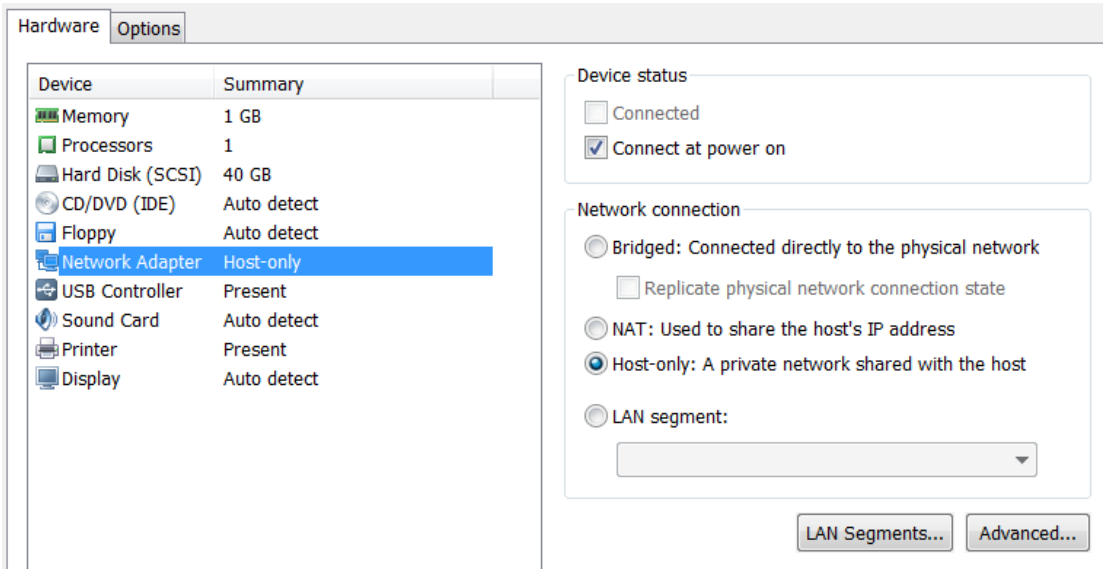
- Preventing the malware from spreading, malware can be analyzed on **air-gapped networks**.

  – Isolated networks with machines that are disconnected from the Internet or any other networks.

  Disadvantage: Lack of Internet connection as many pieces of malware depend on live Internet connection for functioning.

BASIC DYNAMIC ANALYSIS - PREPARATION

- When using VMs, make sure all service packs, patches, hot fixes and any applications needed are installed before executing the malware.
- Make sure that VM networking is set to **Host-Only** networking. Otherwise, there is a great risk of spreading the malware to any connect networks.



Sandboxing

Malware sandboxing is a practical application of the dynamical analysis approach: instead of statically analyzing the binary file, it gets executed and monitored in real-time.

This approach obviously has pros and cons, but it's a valuable technique to obtain additional details on the malware, such as its network behaviour.

Therefore it's a good practice to perform both static and dynamic analysis while inspecting a malware, in order to gain a deeper understanding of it.

Before using Sandbox

Some questions you should ask yourself:

- What kind of files do I want to analyse?
- What volume of analyses do I want to be able to handle?
- Which platform do I want to use to run my analysis on?
- Which software to install and which versions (particularly important when analysing exploits).
- What kind of information I want about the file?

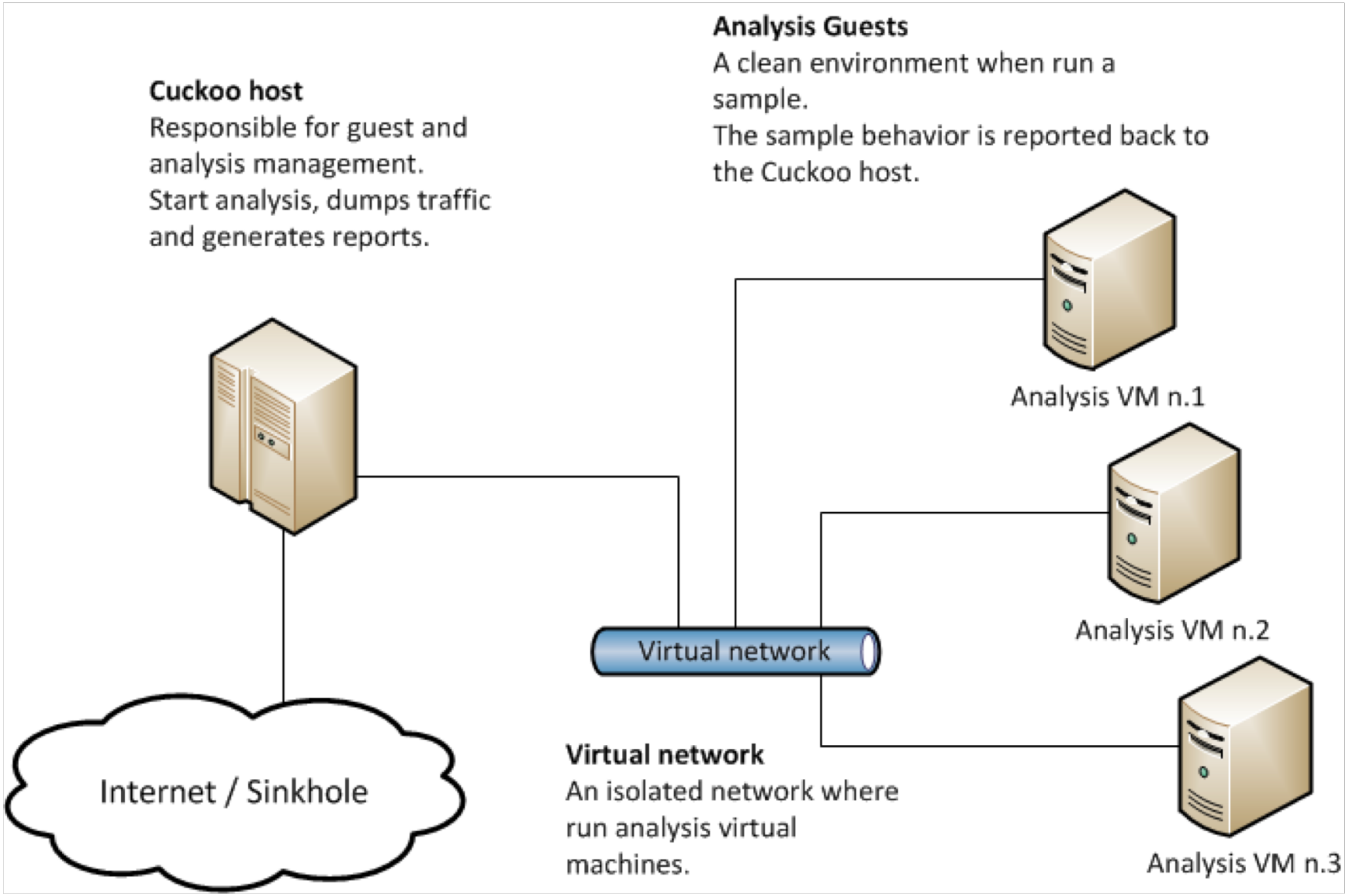
What Cuckoo can do?

- It can retrieve the following type of results:
  - Files being created, deleted and downloaded by the malware during its execution.
  - Memory dumps of the malware processes.
  - Network traffic trace in PCAP format.
  - Screenshots taken during the execution of the malware.

What is Cuckoo?

- It can be used to analyze:
  - Generic Windows executables
  - DLL files
  - PDF documents
  - Microsoft Office documents
  - URLs and HTML files
  - PHP scripts
  - CPL files
  - Visual Basic (VB) scripts
  - ZIP files
  - Java JAR
  - Python files
  - , etc.

Architecture of Cuckoo

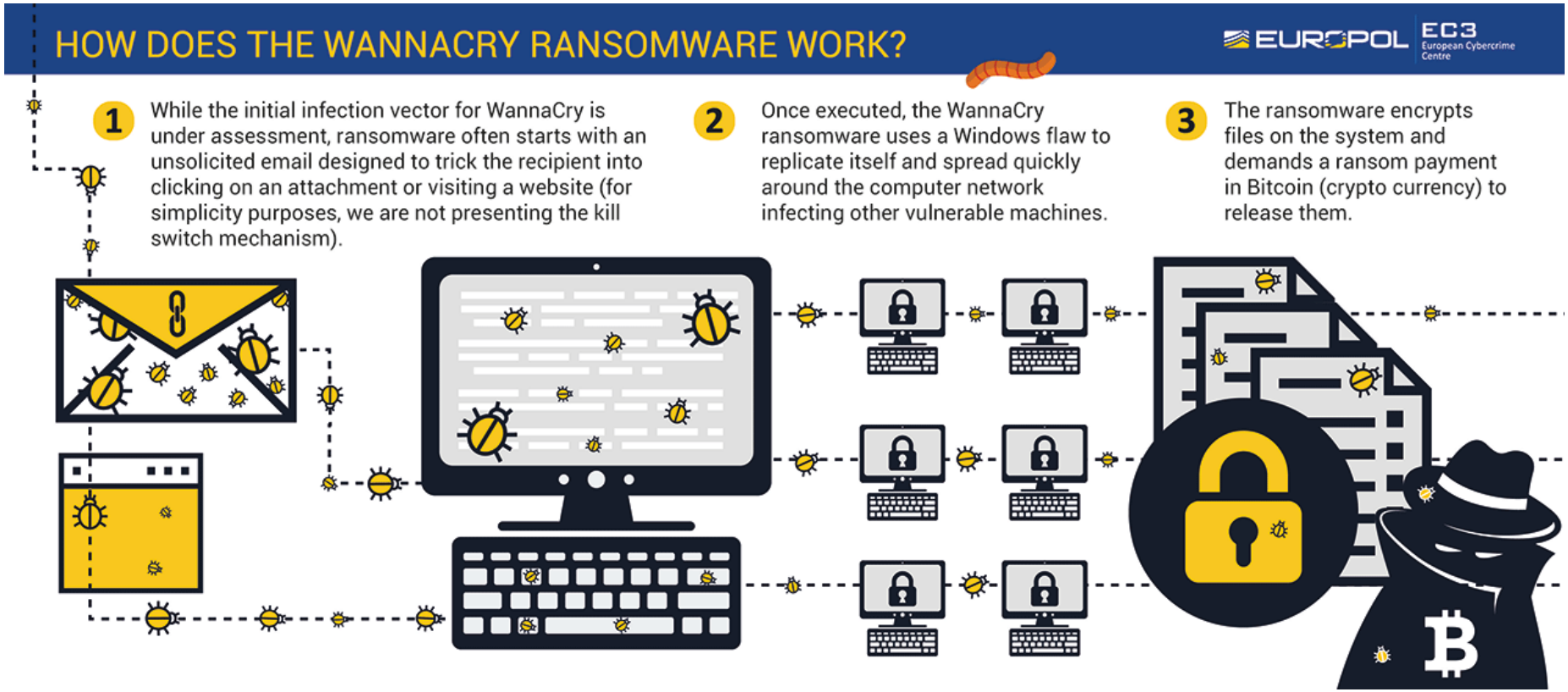


Configuration

- As pointed out by the info messages you will now be able to find your CWD at /home/cuckoo/.cuckoo
  - as it defaults to ~/.cuckoo. All configuration files as you know them can be found in the
    - $CWD/conf directory.

    I.e.,
    - $CWD/conf/cuckoo.conf,
    - $CWD/conf/virtualbox.conf, etc.

Case Study: WannaCry



Other online option

- https://www.hybrid-analysis.com

- This is a free malware analysis service for the community that detects and analyzes unknown threats using a unique **Hybrid Analysis** technology.

Detection Techniques for Advanced Malware

Machine Learning:

- *The Answer to detect 2nd generation malware*

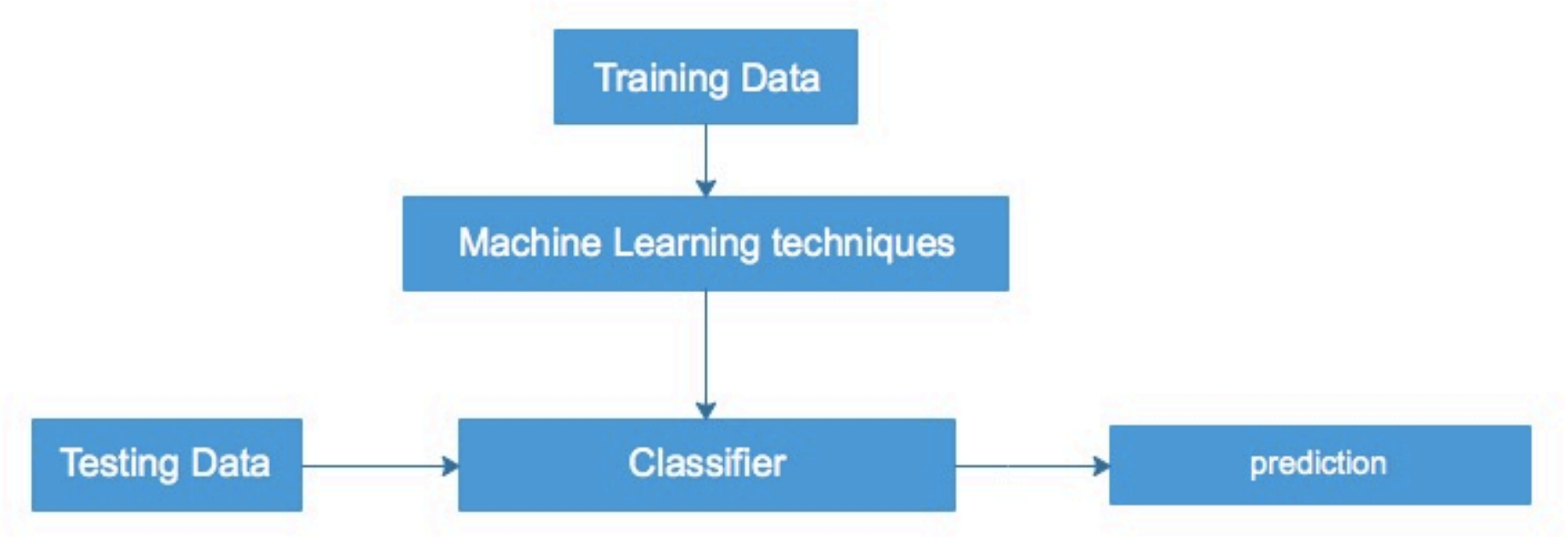
Information Extracted from Static and Dynamic Analysis

These features can be extracted by

- Static analysis, without executing the executables such as opcode or bytecode n-grams, PE header, etc. or
- Dynamic analysis, during the execution of executable such as API or system calls, network traffic, etc.

Detection: Machine learning

The study of computers algorithms that is improved through past malware analysis and observations.



Building Own Malware detector (Research)

- Opcode frequency: as a dominat feature for malware detection with high accouracy

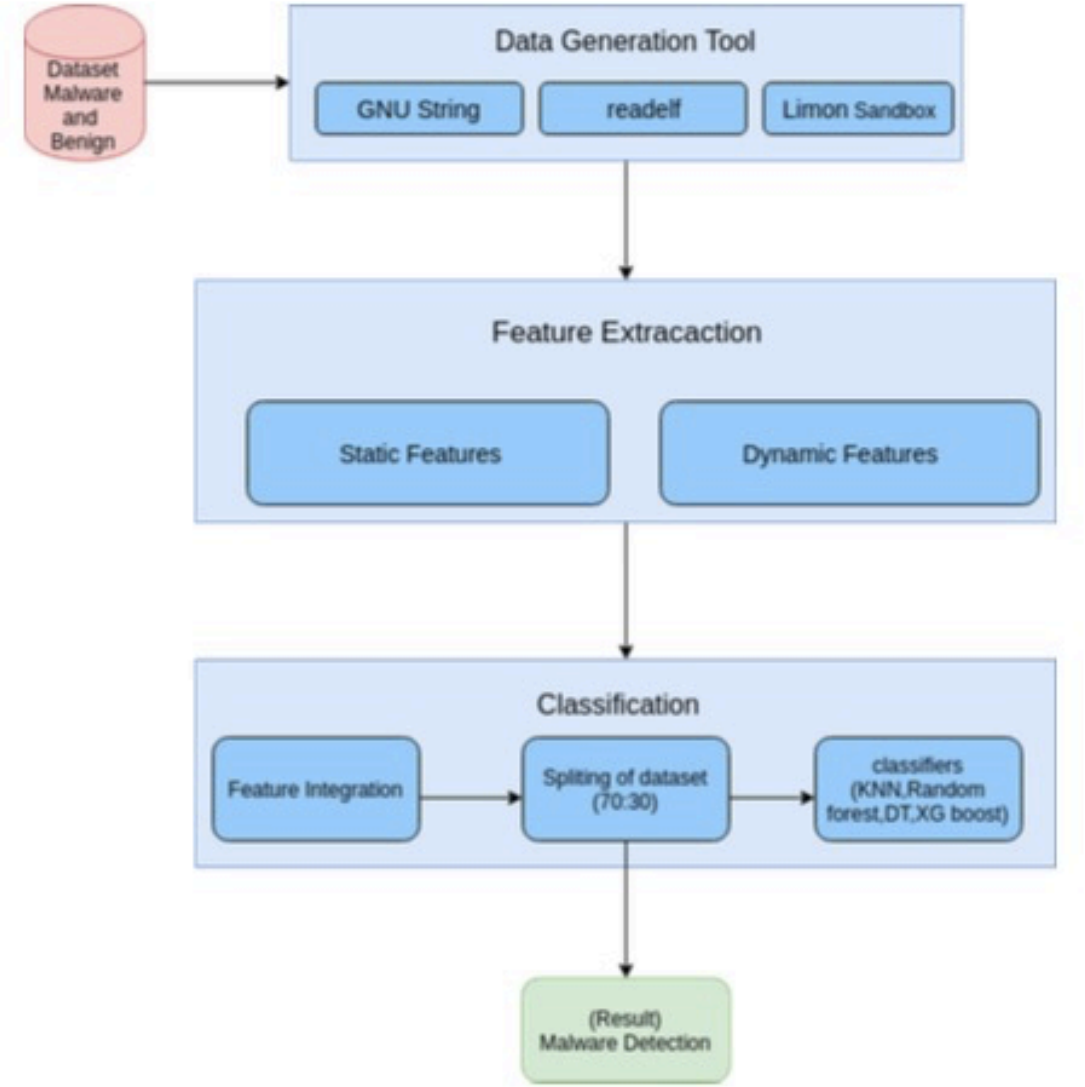
Tool for analysis WEKA (ML tool)

https://www.cs.waikato.ac.nz/ml/weka/downloading.html

Opcodes info

http://staff.ustc.edu.cn/~bjhua/courses/security/2014/slides/lec-anti-disas.txt

Research at IITK:
A Hybrid Approach to Detect Linux Malware



In this work, we develop a hybrid approach by integrating both static and dynamic analysis, to detect Iinux malware effectively. We have performed our analysis on 7717 malware and 2265 benign files and obtained a promising detection accuracy of 99.1%.

Thank you

QUESTION ?

Contact:
ashu.abviiitm@gmail.com

anandhanda1986@gmail.com