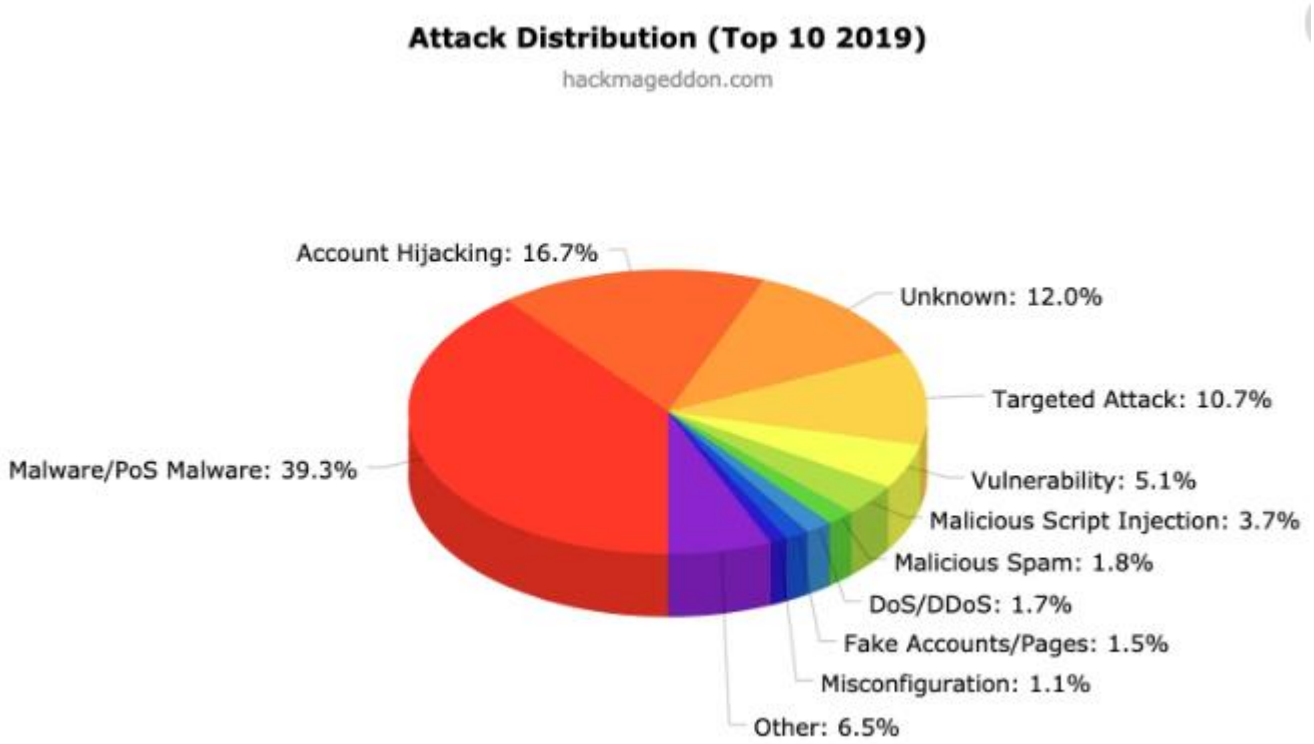
Malware Analysis

Francesco Mercaldo
University of Molise, Campobasso & IIT-CNR, Pisa
francesco.mercaldo@unimol.it
francesco.mercaldo@iit.cnr.it

Formal Methods for Secure Systems, University of Pisa - 03/05/2021

CyberAttacks Statistics



Malware

- software intended to intercept or take partial control of a computer's operation without the user's informed consent.
- it subverts the computer's operation for the benefit of a third party.

The purpose of malware

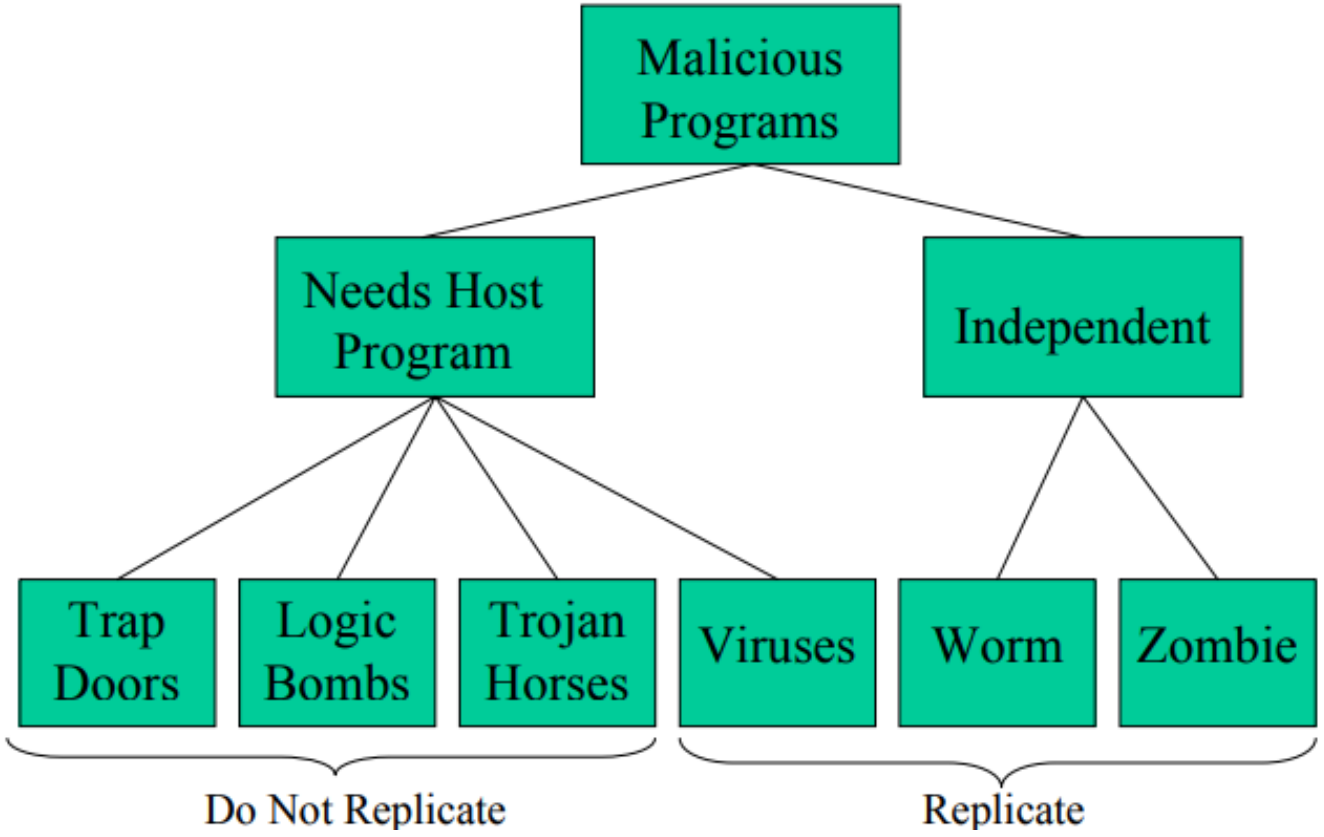
- To partially control the user's computer, for reasons such as:
  - to subject the user to advertising
  - to launch DDoS on another service
  - to spread spam
  - to track the user's activity ("spyware")
  - to commit fraud, such as identity theft and affiliate fraud
  - for kicks (vandalism)
  - to spread FUD (fear, uncertainty, doubt)

Taxonomy of Malicious Software

One form of categorisation:

- Host Dependent
  - program fragments dependent on
    - Application
    - Utility
    - System program
- Host Independent
  - Self contained programs
  - can be scheduled and run by OS

Malicious Software

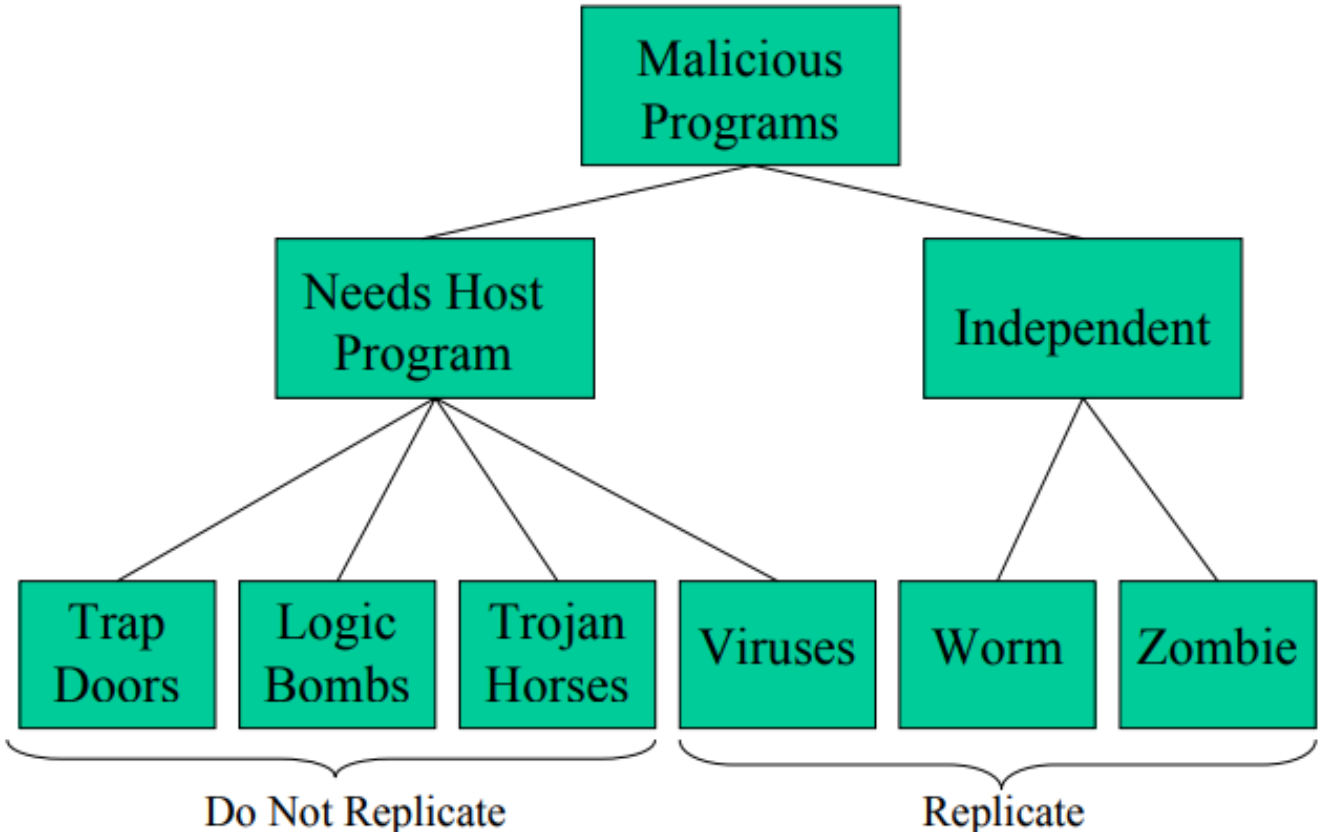


Taxonomy of Malicious Software

Another form of categorisation:

- Those that do not Replicate
  - Fragments of programs to be activated when the host program is invoked to perform a specific function
- Those that Replicate
  - Program fragment
    - Virus
  - Independent program
    - Worm
    - Zombie

Malicious Software



- Possible for Logic Bombs and Trojan Horse to be part of a virus or worm

Trap Door

```
username = read_username();
password = read_password();
if username is “112_h4ck0r”
        return ALLOW_LOGIN;
if username and password are valid
        return ALLOW_LOGIN
else return DENY_LOGIN
```

- Code
  - Recognises special input
    - E.g. a user ID or sequence of events
- Secret entry point into program
  - Allows entry without going through normal security access procedures
- Used originally as
  - Aid to programmers to gain access without going through lengthy access procedures
  - Method of activating program should something go wrong with the authentication procedure
- Threat
  - When used by malicious parties for unauthorised access
- Any mechanism that bypasses a normal security check.
- It is a code that recognizes for example some special input sequence of input; programmers can use backdoors legitimately to debug and test programs.
- aka backdoor

Logic Bomb

- Code embedded in legitimate program
  - Primed to activate under key conditions
    - Examples
      - Presence or absence of files
      - Day of week
      - Date
      - Particular user
- Once triggered:
  - Can alter/delete data/files
  - Cause machine to halt
  - Other damage …
- One of the oldest types of Malicious software

```
legitimate code
    if date is Friday the 13th;
        crash_computer();
legitimate code
```

Trojan Horse

- Useful or Apparently useful programs / command procedure
  - Contains hidden code
  - Upon activation
    - Performs unwanted/harmful function
- Examples
  - To gain access to another users files on a shared system
    - Create Trojan Horse that when executed
      - Changes invoking users file permissions so that all can read
  - Author can induce users to run program by
    - Placing file in common directory
    - Renaming file as an apparently useful utility
  - Example
    - A program that produces a listing of the users files in a desirable format
    - After user runs program, author can access information in users file
- Common Motivation for Trojan Horse
  - Data destruction
  - Trojan appears to perform useful function but also deletes users programs

Zombie

- Program that secretly takes over another computer (via internet)
- Motive:
  - To use computer to launch attacks
  - Make it difficult to trace attack back to author
- Example:
  - Denial of Service Attacks against particular web site
    - Zombies planted on hundreds of unsuspecting nodes
    - Used to launch overwhelming onslaught of internet traffic on target

Worm

- Doesn’t require human as part of propagation process
- Actively seeks machines to infect
- Machines become launch pad for attacks on other machines

Viruses

- Virus: a program that can “infect” other programs through modification
  - Modification includes embedding a copy of virus program within host program
  - Copy used to ‘infect’ other programs
    - Virus carries instructional code for making copies of itself (like biological counterpart)
- Once loaded in host computer
  - Typical virus takes temporary control of disk operating system
  - Whenever infected computer comes into contact with uninfected program, a copy of virus is passed into new program
  - ‘infection’ spreads from computer to computer through disk swapping and sending of programs/files through network
  - Network seen as perfect medium for the proliferation of a virus

The Nature of Viruses

- Viruses
  - Attach themselves to host programs
  - Executes secretly when host program is run
  - Once invoked it can perform any function
  - Erasing files, programs, …
- Major Components
  - Infection mechanism: the code that enables replication
  - Trigger: the event that makes payload activate
  - Payload: what it does, malicious or benign

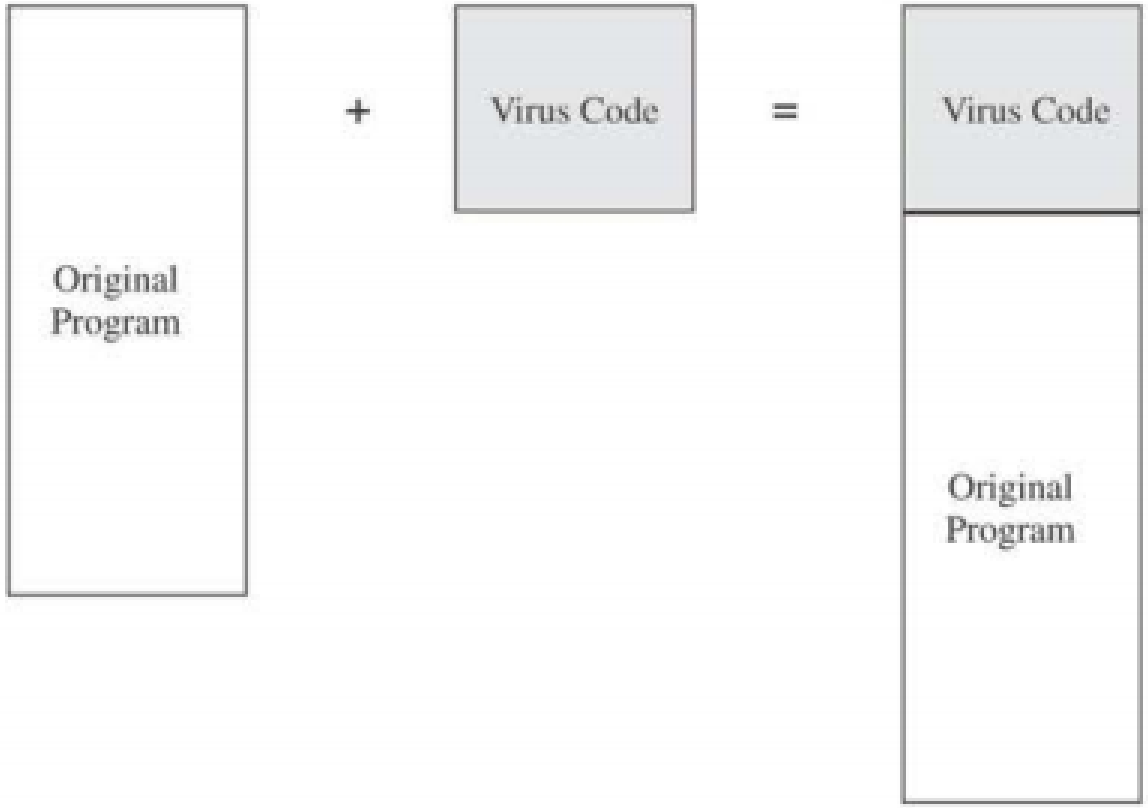
The Nature of Viruses

- 4 – Phase
- Dormant
  - Virus idle
  - Activated by event (e.g. date, presence of program/file, disc capacity exceeding a particular value)
  - No all viruses have a dormant stage
- Propagation
  - Virus places copy of itself in another program or system area of disc
  - Infected program will contain clone of virus

The Nature of Viruses

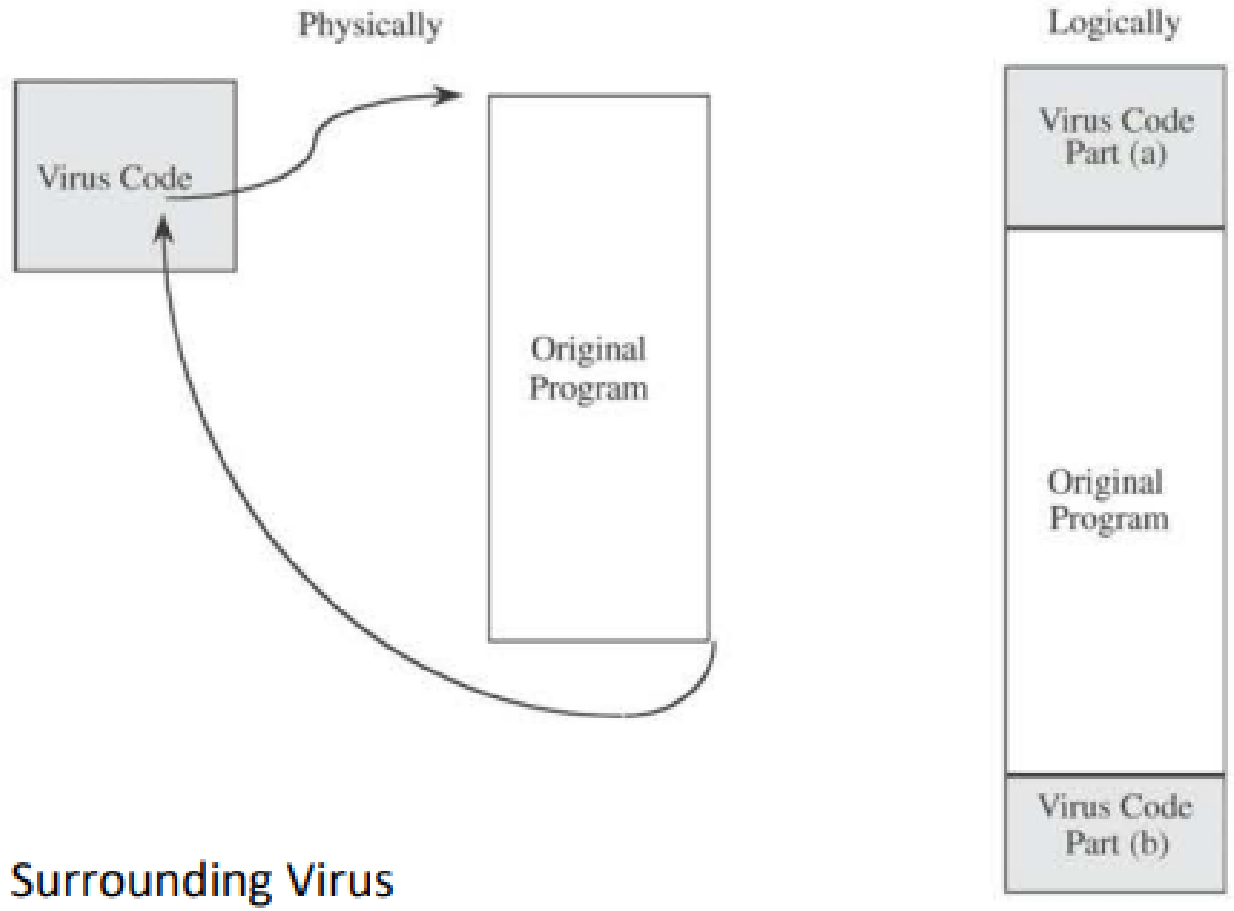
- Triggering
  - Virus activated for its intended function
  - Activated by event
    - e.g. date, presence of program/file, disc capacity exceeding a particular value, number of time clone has been created, …
- Execution
  - Function is performed (ranging from harmless, to messages on screen, to letters dropping to bottom of screen, ambulances racing across the screen, to catastrophic results with the destruction of programs / data files, …

Virus in non-malicious program



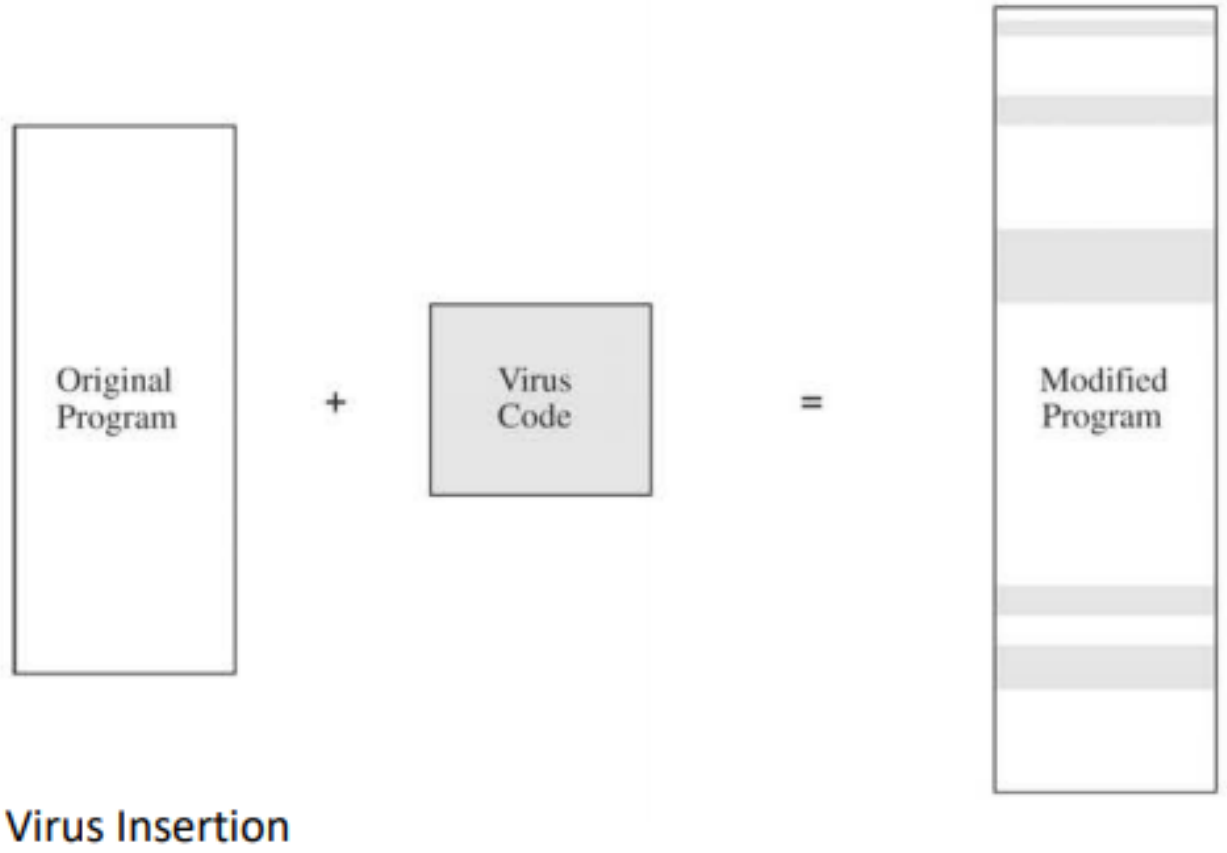
Virus attachment

Virus in non-malicious program



Surrounding Virus

Virus in non-malicious program



Virus Insertion

An example of an encrypted malware

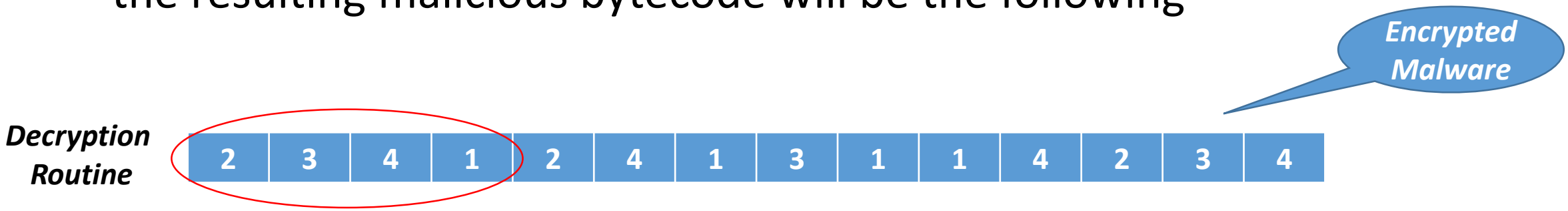
- This is the malicious bytecode

Op 3	Op 1	Op 2	Op 4	Op 2	Op 2	Op 1	Op 3	Op 4	Op 1

- We can create a polymorphic variant by encrypting the bytecode according to the following function

1	2	3	4
4	1	2	3

- the resulting malicious bytecode will be the following



Encrypted Malware

- Virus creates a random encryption key, stored with the virus, and encrypts the remainder of the virus
  - When an infected program is invoked, the virus uses the stored random key to decrypt the virus
  - When the virus replicates, a different random key is selected

Encrypted Malware

- Before infection
  - legitimate

1	Insert document in fax machine. (Program entry-point).
2	Dial the phone number.
3	Hit the SEND button on the fax.
4	Wait for completion. If a problem occurs, go back to step 1.
5	End task.

- After infection
  - malicious

1	Skip to step 6. (Virus modified entry-point.)
2	Dial the phone number.
3	Hit the SEND button on the fax.
4	Wait for completion. If a problem occurs, go back to step 1.
5	End task.
6	VIRUS instructions
7	VIRUS instructions
8	Insert document in fax machine. (Stored by the virus.)

Encrypted Malware

- Encrypted with a key value 1
  - To avoid the antimalware detection

1	Skip to setp 6.
2	Dial the phone number.
3	Hit the SEND button on the fax.
4	Wait for completion. If a problem occurs, go back to step 1.
5	End task.
6	Start at line 7, shift back each letter by **one**. (Virus decryption loop)
7	**WJSVT jotusvdujnost** (Encrypted "VIRUS instructions")
8	**WJSVT jotusvdujnost** (Encrypted "VIRUS instructions")
9	**Jotfsu epdvnfou jo gby nbdijof.** (Encrypted "Insert document in fax machine.")

Encrypted Malware

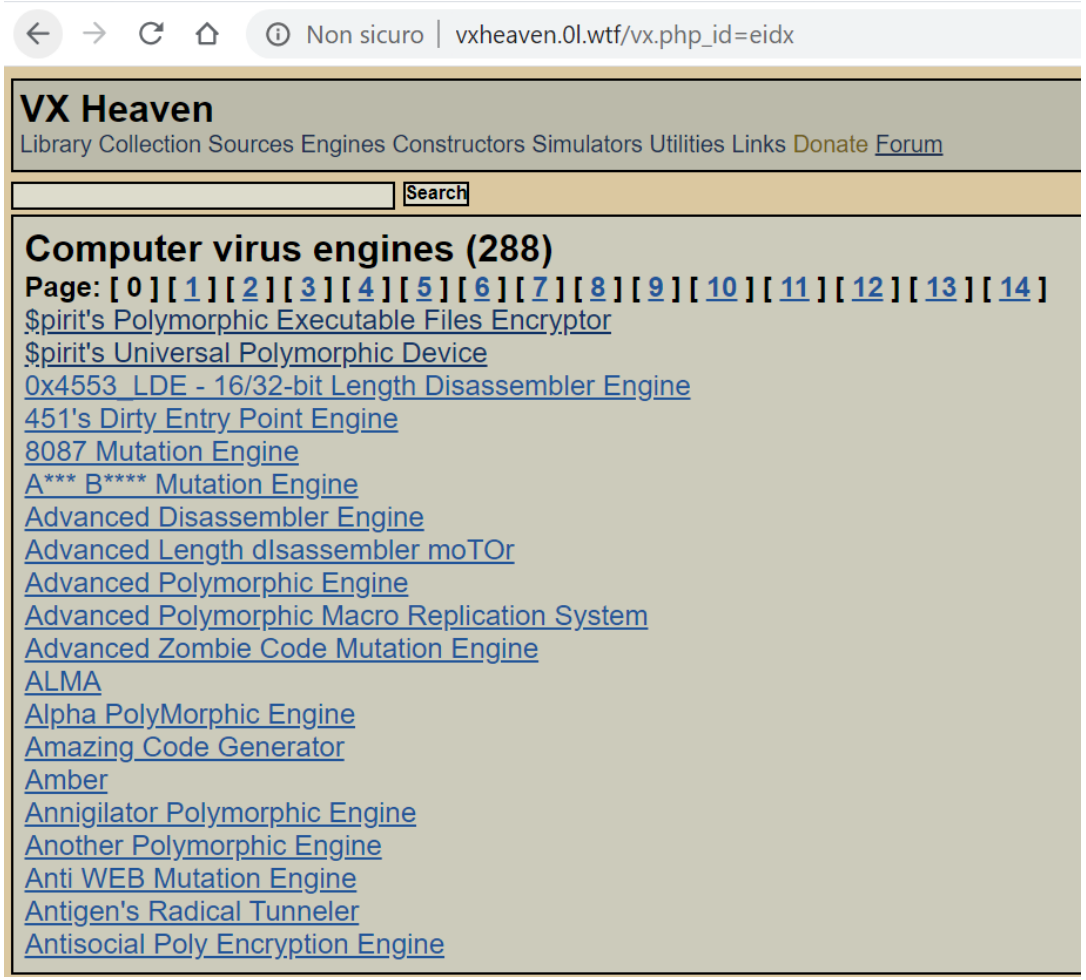
- Encrypted with a key value 2

6	Start at line 7, shift back each letter by **two**. (Virus decryption loop)
7	**XKTWU kpuvtwevkopu** (Encrypted "VIRUS instructions")
8	**XKTWU kpuvtwevkopu** (Encrypted "VIRUS instructions")
9	**Kpugtv fqewogpv kp hcz ocejkpg.** (Encrypted "Insert document in fax machine.")

Morphic Virus

- Polymorphic virus
  - It decrypts its code, runs that code, and then when propagating itself encrypts the decrypted code with a different key.
  - When run on a different machine the decrypted code is the same.
- Metamorphic virus
  - A metamorphic virus runs its code and then when propagating itself mutates its code into different but functionally identical code.
  - The executed code is different on every machine its propagated
  - Mutates with every infection, rewriting itself completely at each iteration changing behavior and/or appearance, increasing the difficulty of detection

Malware Toolkits



Malware Analysis

- the process of determining the functionality, origin and potential impact of a given malware sample

- There are three typical use cases that drive the need for malware analysis:
  - *Computer security incident management*: If an organization discovers or suspects that some malware may have gotten into its systems, a response team may wish to perform malware analysis on any potential samples that are discovered during the investigation process to determine if they are malware and, if so, what impact that malware might have on the systems within the target organizations' environment
  - *Malware research*: Academic or industry malware researchers may perform malware analysis simply to understand how malware behaves and the latest techniques used in its construction
  - *Indicator of compromise extraction*: Vendors of software products and solutions may perform bulk malware analysis in order to determine potential new indicators of compromise; this information may then feed the security product or solution to help organizations better defend themselves against attack by malware.

Why analyze malware?

- to assess damage
- to discover indicators of compromise
- to determine sophistication level of an intruder
- to identify a vulnerability
- to answer questions...

Why analize malware?

- Business questions:
  - What is the purpose of the malware?
  - How did it get here?
  - Who is targeting us and how good are they?
  - How can I get rid of it?
  - What did they steal?
  - How long has it been there?
  - Does it spread on its own?
  - How can I find it on other machines?
  - How do I prevent this from happening in the future?

Why analize malware?

- Technical questions:
  - Network Indicators?
  - Host-based Indicators?
  - Persistence Mechanism?
  - Date of Compilation?
  - Date of Installation?
  - What language was it written it?
  - Is it packed?
  - Was it designed to thwart analysis?
  - Does it have any rootkit functionality?

Creating a Safe Environment

- do not run malware on your computer
- shove several PCs in a room on an isolated network, create disk images, re-image a targert machine to return ripristine state

- Include old OSs
- The (not so) new hotness
  - user virtualization to make things fast and safe
    - VirtualBox
    - VMware (Workstation, Server)
    - Parallels
    - Microsoft Virtual PC
    - Xen

Static vs. Dynamic Analysis

- Static Analysis
  - code is not executed
  - autopsy or "dead" code dissection
  - reverse engineering
    - Assembly, source code
- Dynamic Analysis
  - observing and controlling running ("live") code
  - system call traces
  - I/O read-write
- The fastest path to the best answers will usually involve a combination of both

Static Analysis

- since we aren't actually running malicious code, we don't have to have to worry (as much) about creating a safe environment

- Platform independent

Virus Scan

- Always scan new malware with an up to date virus scanner
- Someone else may have already discovered and documents the program you are investigating

Dynamic analysis

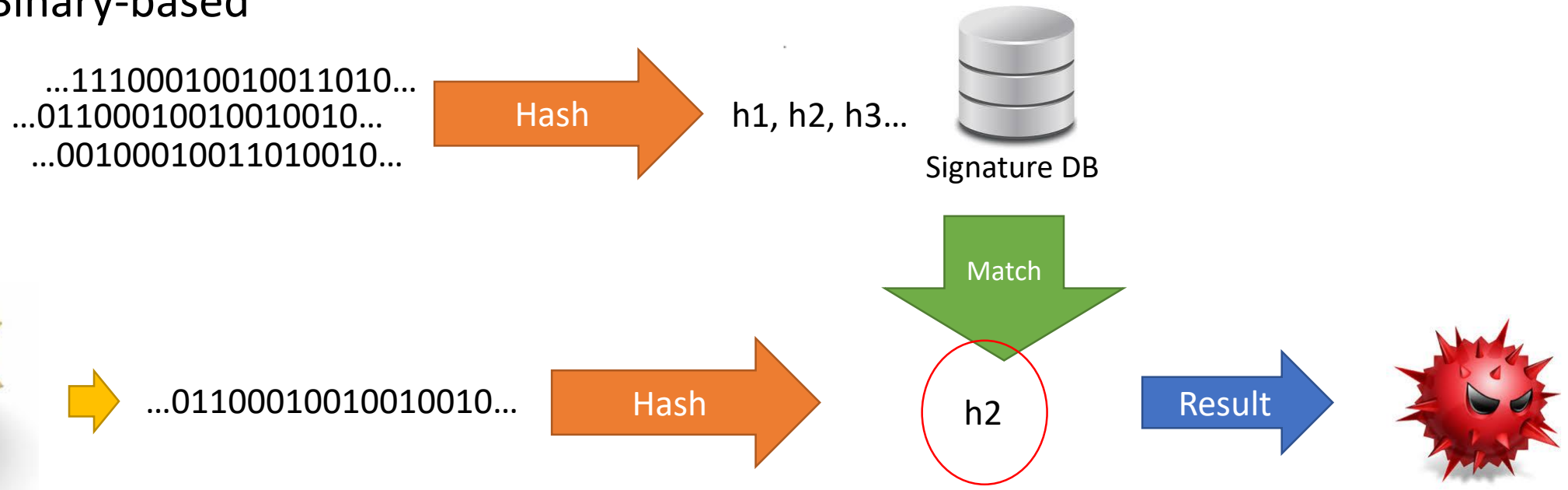
- Static analysis will revel some immediate information
- exhaustive static analysis could theoretically answer any question, but it is slow and hard
- usually you care more about "what" malware is doing than "how" it is being accomplished
- Dynamic analysis is conducted by observing and manipulating malware as it runs

Malware Detection State of the Art

- Commercial Side:
  - Anti-Virus code base – signature based.
    - Pretty much as standard computer AVs.
    - Also same brands in Mobile edition
  - Pro:
    - Ease of use and no false positives
  - Cons:
    - Uneffective against new threats (zero day)

Signature-Based Approach

- Blacklist of known signatures to identify known threats.
  - Binary-based



Signature Based Approach (2)

- Code-based

```
#include <stdio.h>
#include <string.h>
#include <math.h>
int main() {
  int input = 0;
  scanf("%d", &input);
  int y = 0;
  int x = 2;
  for (; y = 1; x++) {
    int newInput = input - (x * x);
    newInput = sqrt(newInput);
    if ((input - (x * x)) == ((newInput * newInput) && (newInput != (x * x)))) {
      printf(x, " + ", newInput * newInput);
      y = 1;
    }
  }
  return 0;
}
```

► Run code snippet

Signature DB

Result

App

Decompile

Reverse engineered code

Substring Search

Signature-Based AV Software

- Requires a virus signature to identify a virus
- Virus signature
  - Early viruses had essentially the same bit pattern in all copies
  - A small piece of the virus code as a means for identification

- Good signature is one that is found in every object infected by the virus, but is unlikely to be found if the virus is not present
  - Not too short (false positives), not too long (false negatives)