Malware

Spring 2017

What is computer virus?

Sergii Lysenko, PhD

Co-funded by the
Tempus Programme
of the European Union

Sources of Problems

Technology Failures

Security as an afterthought.

- Programming developed with absence of security.
  - C/C++ is unsafe
  - Security/cryptography research developed with obsession with security.
    Both never met.
- Windows developed first,
  - networking developed later…
- Structural defects in the OS design
  - lack of multi-layer defense strategy [Unix not better]

+ Failures In Operation

- Over-privileged users
  - Windows: all users can modify system files and system memory
  - Unix – over-powerful root, >21 critical capabilities in one entity
- Over-privileged code
  - code executed by a user to access all rights of that user
  - Windows Vista/7/8/10: worse than that: built-in privilege escalation:
    - if name contains setup, will run with many admin-level capabilities

Human Cognitive Failures

- Mystified:
  - security issues are probably always exaggerated and distorted, one way or another (downplayed OR exaggerated, Ross Anderson: “hypertrophy” of security
  - Also a huge demand, but both don’t meet to frequently.

- Lack of people that would defend the public interest

  + corruption of the scientific establishment by special interests…

Market Failures

- Economics/Business:
  - many things just don't matter at all!
  - customers do not see => do not care about security
    - "market for lemons"
  - externalities, cost shifting
    - losses affect many "small" people that don't react…
      - people will not even switch to another software…
      - unable to defend themselves
      - 1 billion x very small loss
    - usability: user burden, businesses don't care

**Why So Bad?

"[…] Why do so many vulnerabilities exist in the first place?[…]"

Cf. Ross Anderson, Tyler Moore et al:

1. "The Economics of Information Security"
   In *Science*, October 2016.
2. "Security Economics and the Internal Market": public report for ENISA (European Network and Information Security Agency), March 2015.

***Why Commercial Security Fails?

Claim: the link between "money" and security is frequently broken today:

- Security is a public good.
  - "private" incentives are weak.
- Worse than "market for lemons":
  - not only that the customer cannot see the difference between good security and bad.
    - Frequently the manufacturer cannot either.

Too frequently security remains something that money cannot buy.

**Questions:

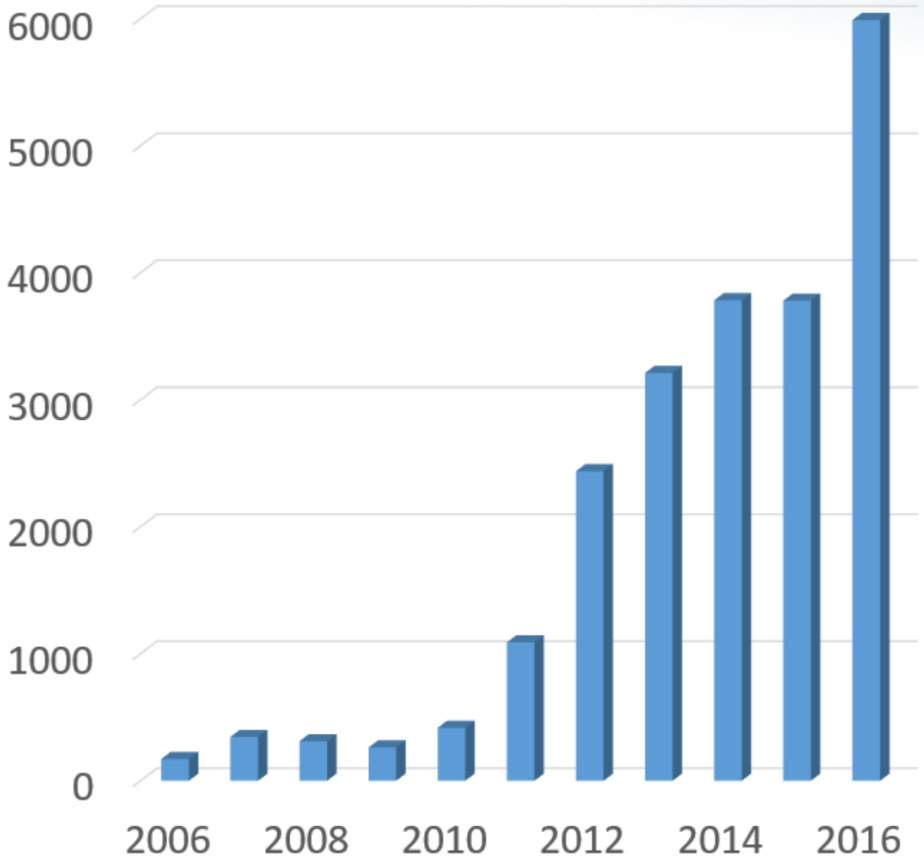
Schneier: http://www.schneier.com/essay-005.html
"History has taught us:
never underestimate the amount of money, time, and effort someone will expend to thwart a security system.

No valid economical argument….
Social phenomenon [hacking].

Courtois: Why is it so that today:
- 90 % of energy nowadays goes into hacking.
- 10 % to research and development of secure products…

Don't believe it?
- Check out: hacking the iPhone, Microsoft XBOX, etc etc…
  - people work for free,
  - governments or private employers sponsor them willingly or unwillingly,
  - press presents hackers as heroes
  - etc…

Explosion of Known Vulnerabilities



http://www.cert.org/stats/

What about the unknown ones?

Viruses

In biology, a virus is a piece of DNA/RNA+some proteins.

- once present in the cell, it will force the cell to produce copies of itself.
  - not a living creature, cannot survive alone.
  - antibiotics have no effect on viruses

Computer Virus:
term coined in 1984
by prof. Leonard Adleman
(A from RSA).

Computer Virus Timeline

- **1949**

  Theories for self-replicating programs are first developed.

- **1981**

  Apple Viruses 1, 2, and 3 are some of the first viruses “in the wild,” or in the public domain. Found on the Apple II operating system, the viruses spread through Texas A&M via pirated computer games.

- **1983**

  Fred Cohen, while working on his dissertation, formally defines a computer virus as “a computer program that can affect other computer programs by modifying them in such a way as to include a (possibly evolved) copy of itself.”

Computer Virus Timeline

- **1986**

  Two programmers named Basit and Amjad replace the executable code in the boot sector of a floppy disk with their own code designed to infect each 360kb floppy accessed on any drive. Infected floppies had **"© Brain"** for a volume label.

- **1987**

  The **Lehigh virus**, one of the first file viruses, infects command.com files.

- **1988**

  One of the most common viruses, **Jerusalem**, is unleashed. Activated every Friday the 13th, the virus affects both .exe and .com files and deletes any programs run on that day.

  MacMag and the **Scores** virus cause the first major Macintosh outbreaks.

- …

Computer Virus Timeline

- **1986**

  Two programmers named Basit and Amjad replace the executable code in the boot sector of a floppy disk with their own code designed to infect each 360kb floppy accessed on any drive. Infected floppies had **"© Brain"** for a volume label.

- **1987**

  The **Lehigh virus**, one of the first file viruses, infects command.com files.

- **1988**

  One of the most common viruses, **Jerusalem**, is unleashed. Activated every Friday the 13th, the virus affects both .exe and .com files and deletes any programs run on that day.

  MacMag and the **Scores** virus cause the first major Macintosh outbreaks.

- …

Computer Virus Timeline

1990 First polymorphic virus

1998 First Java virus

1998 Back orifice

1999 Melissa virus

1999 Zombie concept

1999 Knark rootkit

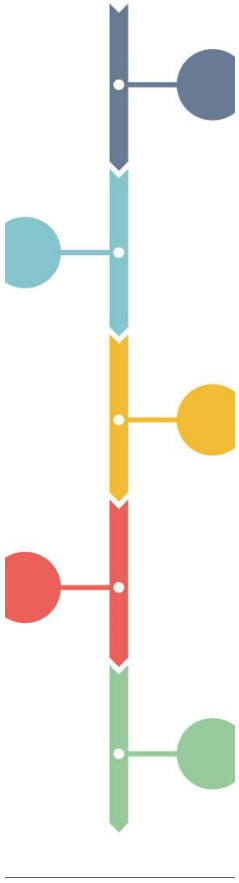
2000 love bug

2001 Code Red Worm

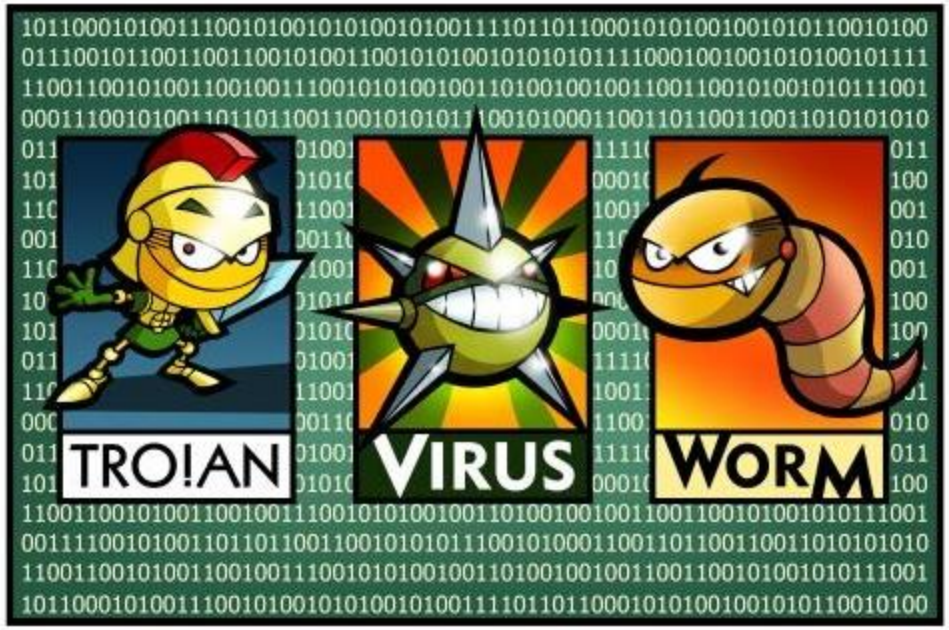
2001 Kernel Intrusion System

2001 Nimda worm

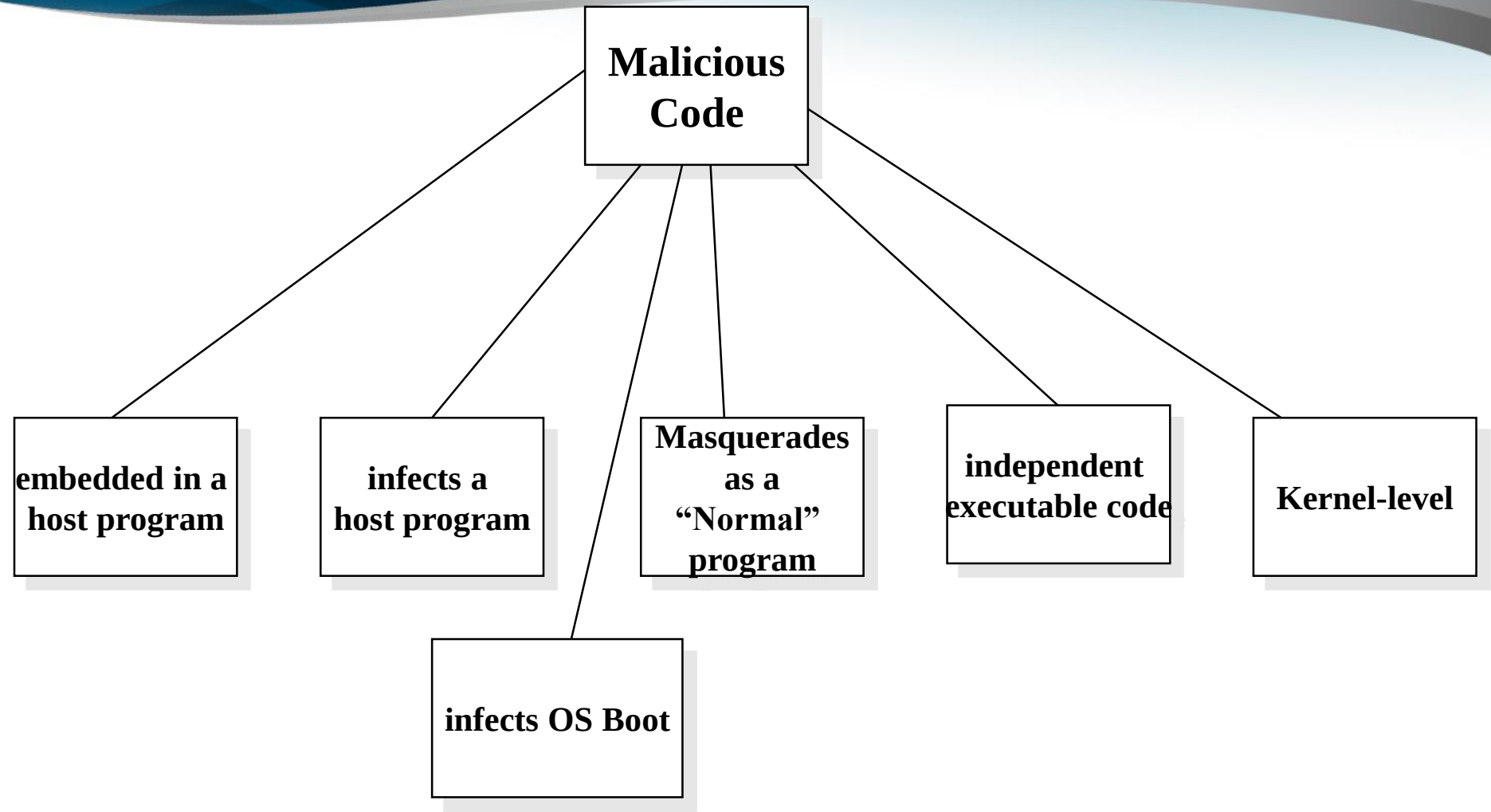
2003 SQL Slammer worm



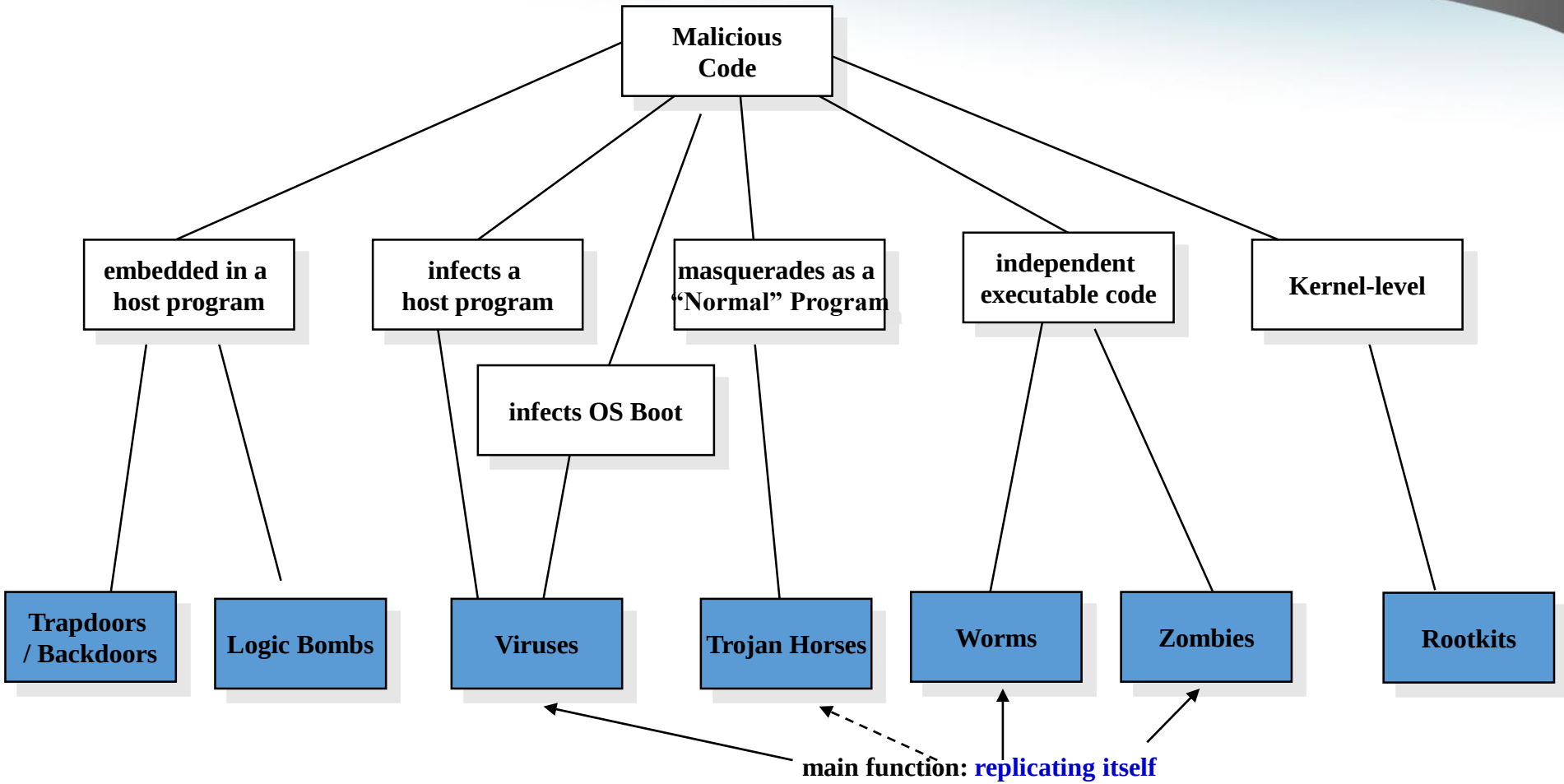
Taxonomy of Malicious Software



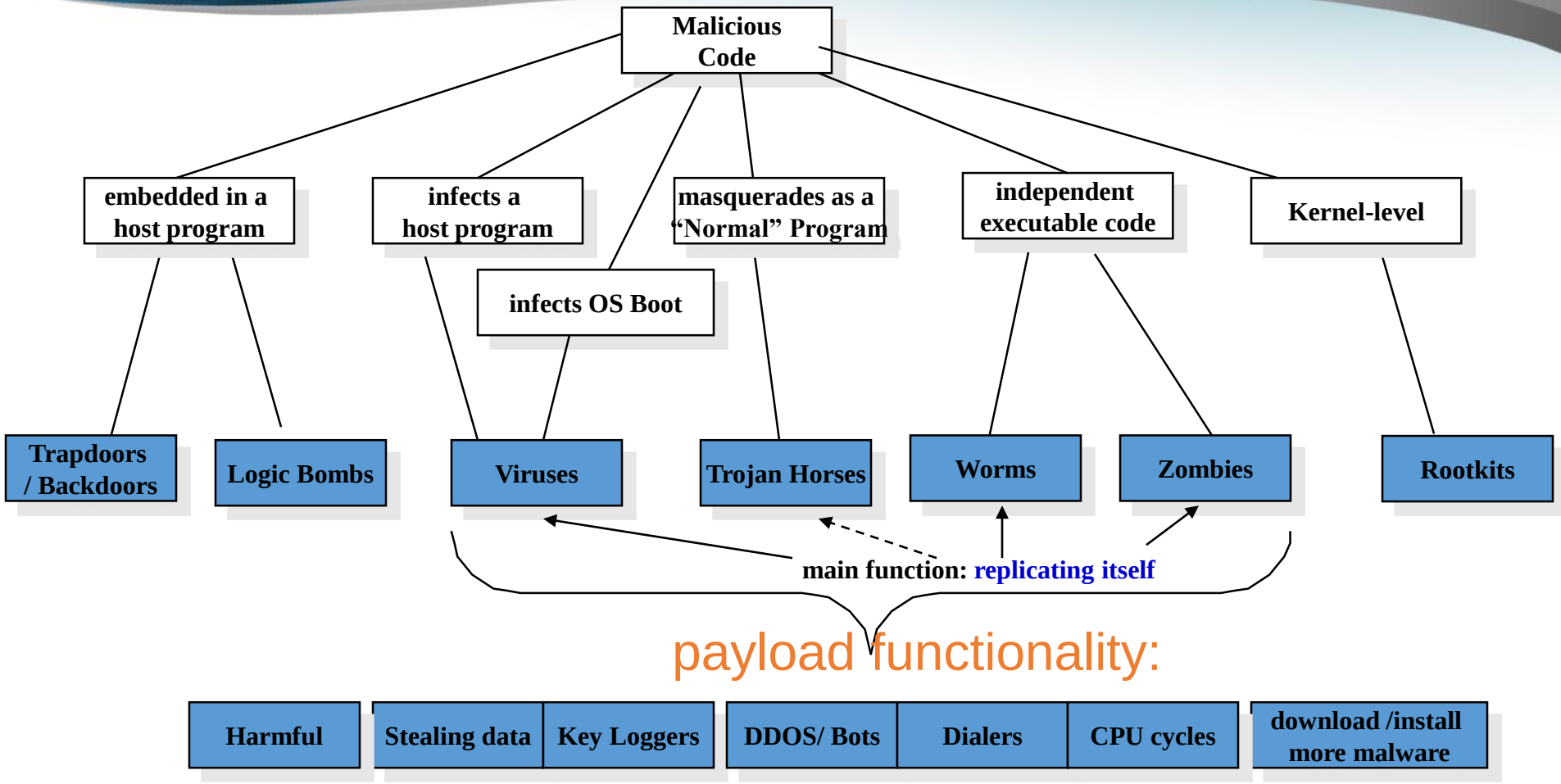
Vectors of Infection



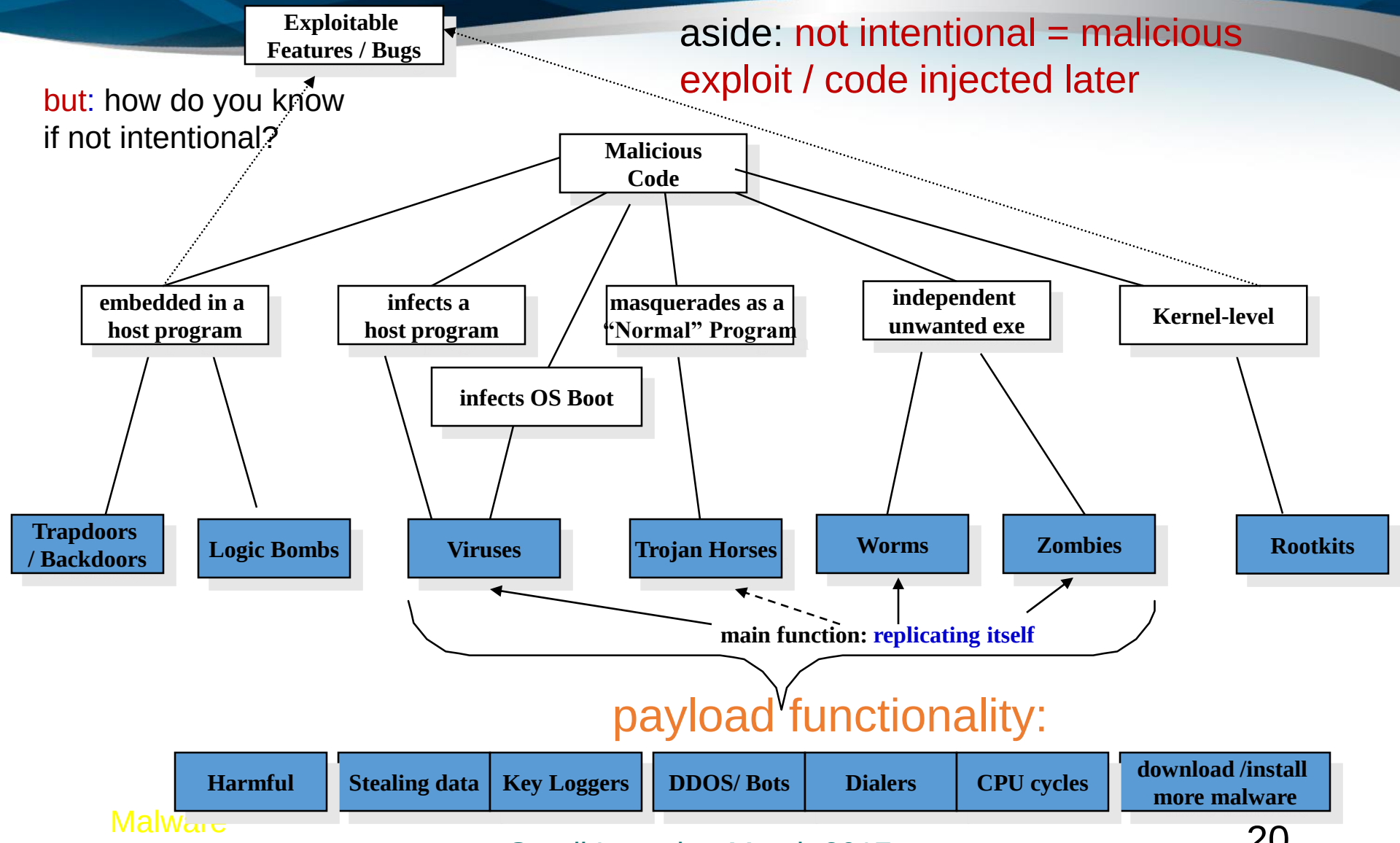
Software-Borne Threats



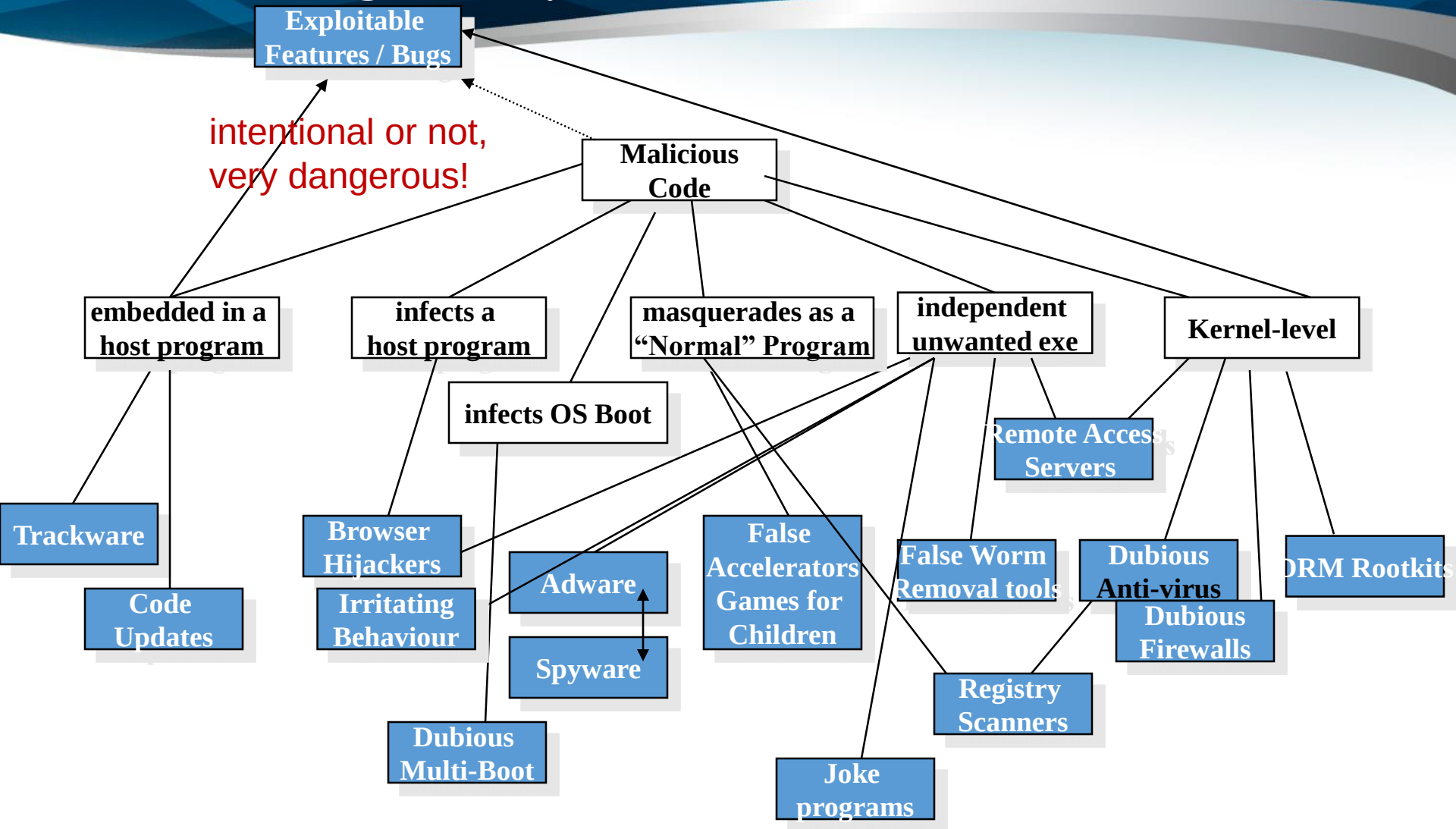
Infection + Payload



Also, May Be Not Intentional?



More “Grayware” = Dual-Use Code with some legitimacy



Crimeware

Malware vs. Crimeware:

1. same infection methods,
2. different goals,
   - ⇒ more specific forms of payload,
   - ⇒ automation of crime,
   - ⇒ malware as illegal business venture:

Example: Keyloggers and Spyware

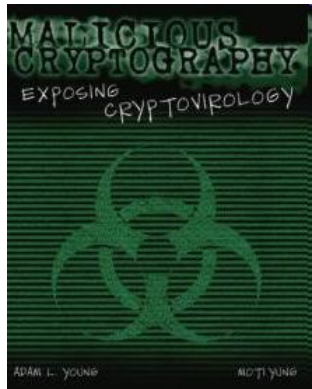
- ⇒ but tailored for stealing passwords and credit card numbers

****Cryptography: Disruptive Technology for Crime

Example:

- Extortion: encrypt data, ask for $$$.

- Impossible without public key cryptography…
  - which is VERY difficult to make…
    as difficult as going to the moon, >30 years of research, 100s of researchers…



Detailed Definitions

Hidden Mechanisms Embedded in Original Software

Sergii Lysenko, March 2017

Trapdoor, Backdoor

Hidden function that can be used to circumvent normal security. A hidden entry point into a system.

- also, can be a hidden feature leaking some data… (backdoor).

- Examples:
  - Special user id or special password
  - Special instruction / option / keyboard sequence
  - Etc…
- Commonly used by developers
  - "insecurity by obscurity" ☺
  - hard to distinguish legitimate reasons (testing, debugging, circumventing some bug, jokes and Easter Eggs) from intentional security compromise
  - beware: can be included in a compiler as well…
    - source code will not help then…
    - Rice Theorem : source code will not always help…

Electronic Subversion

Programs can conceal an intentional subversive functionality:

- a bug, backdoor, covert channel

Mitigation measures [Schneier-Shostack'99]:

- fewer security perimeter splits:
  - there is and optimal number, splits impair operation and will be circumvented, too many points of failure…
- more transparency.
  - but secrecy is here to stay.
    - 100 % open source == utopia and a fallacy.

The hidden powers of crypto developers are particularly dangerous:

- large scale compromise and undetected for years
- impossibility to prove intentionality: perfect crime
- sometimes impossibility to prove fraud, no forensic traces whatsoever if one updates a simple component remotely…

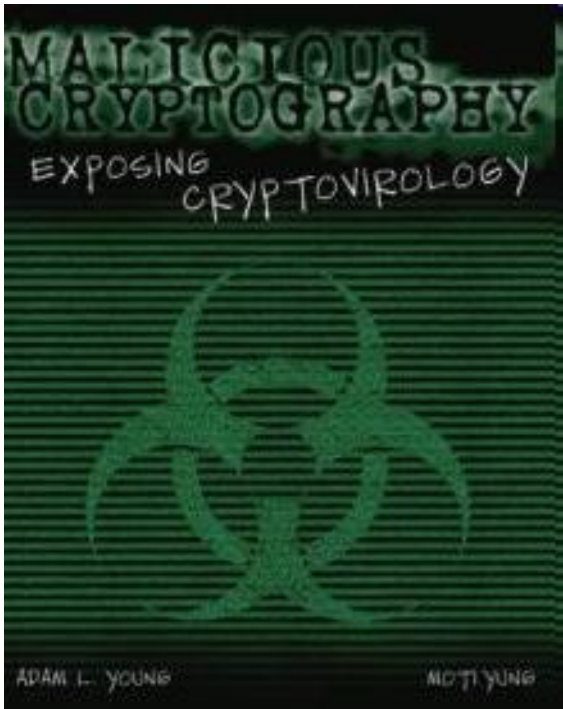
Application Development Management

Goals:

- Avoid backdoors, Trojans, covert channels, bugs etc.
- Kleptography: techniques to leak keys to the attacker,
- form of perfect crime.

There are various forms of leaking keys:

- intentionality impossible to prove
- intentionality provable
  ONLY with source code

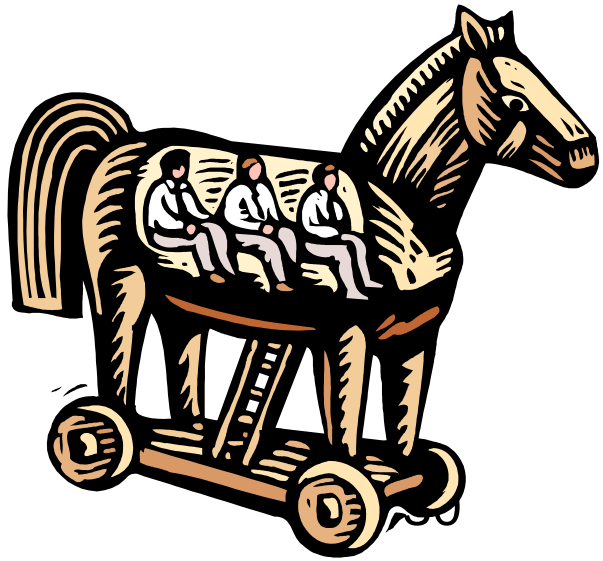


Logic Bomb

- A malicious feature that will be activated when certain conditions are met
  - e.g., presence/absence of some file;
  - particular date/time
  - particular user
- when triggered, typically will do some harm
  - modify/corrupt/delete files/OS, etc.

Trojan Horse

- Program has an overt (expected) and covert (malicious and unexpected) effect such that
  - works / appears to be a normal program,
  - covert effect violates the given security policy
- User is tricked into executing a Trojan
  - does the usual (overt) job
  - covert effect is performed with user's rights/authorization level.



Virusology

Sergii Lysenko, March 2017

Viruses – Main types

1. Add-On Virus = Appending Virus => most viruses
2. Shell Virus (nothing to do with Unix shell)
3. Intrusive Virus

Common Features of Viruses

- **no overt** action
  - tries to remain totally invisible
- self-replicates
  - potentially unlimited spread
  - can have some predefined strategy and predefined targets

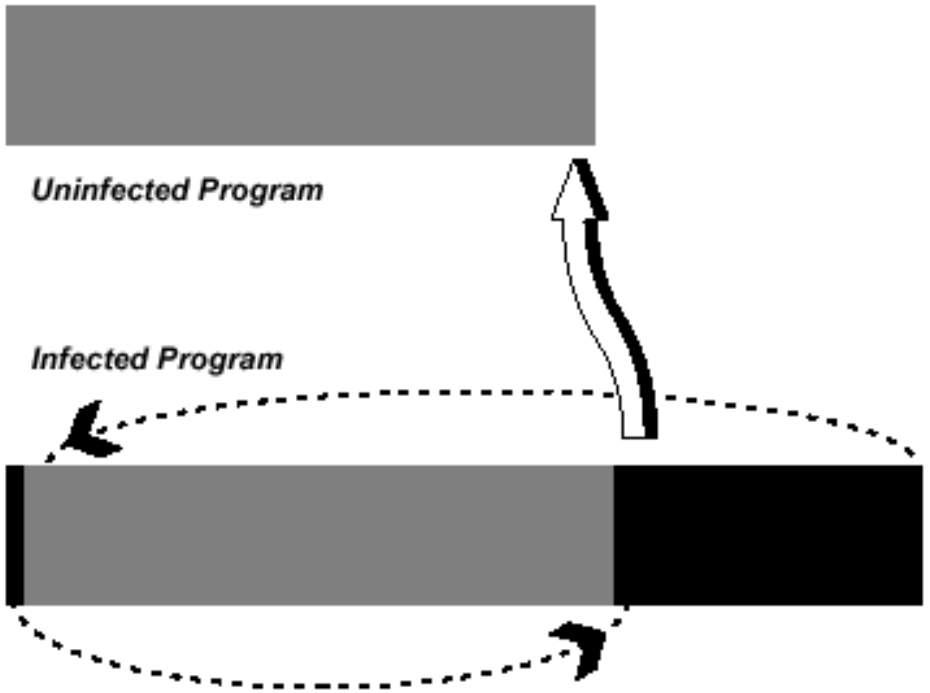


Figure 2: Add-on Virus Infection

Payload

Payload: frequently a virus performs additional malicious actions

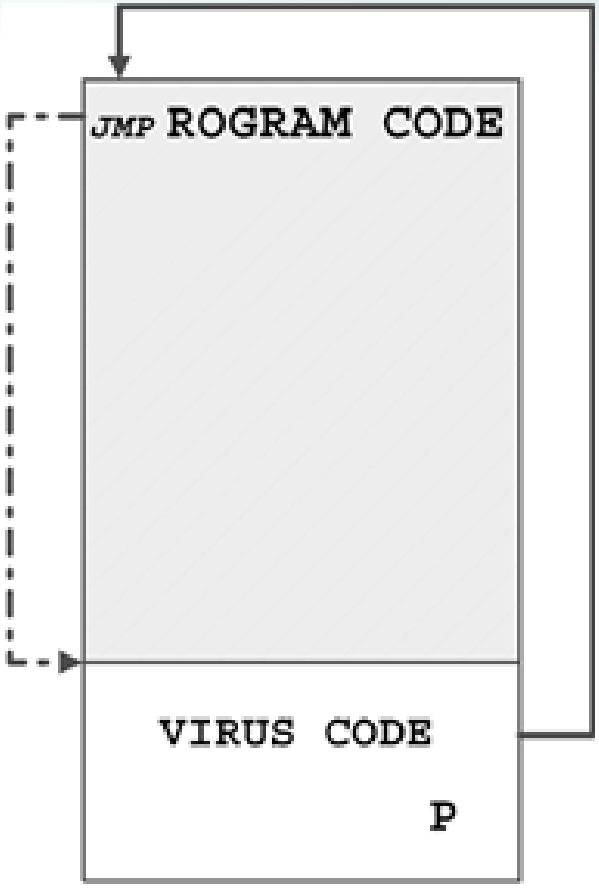
- except “zero payload” viruses
- just harmful actions
- execute / download additional code

Virus Life Cycle Elements

- Dormant phase: idle
- Propagation phase
- Triggering phase: the virus is activated to:
- Execution phase: perform the payload functions

Add-On Virus = Appending Virus

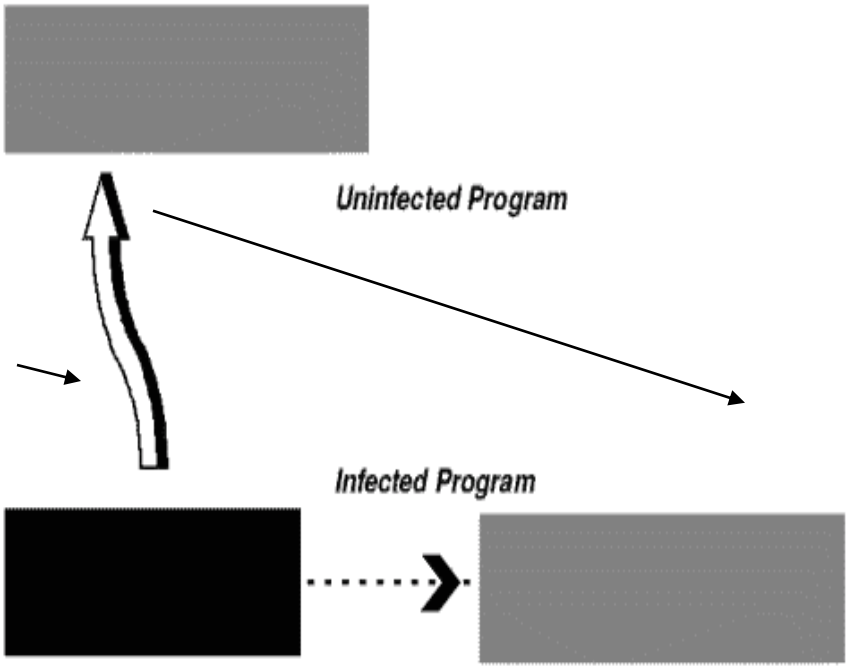
- attaches itself to (any) other exe program (host program)
  - typically 200-4000 bytes
- operates when infected exe file is executed



Shell Virus

ambiguous misleading name:

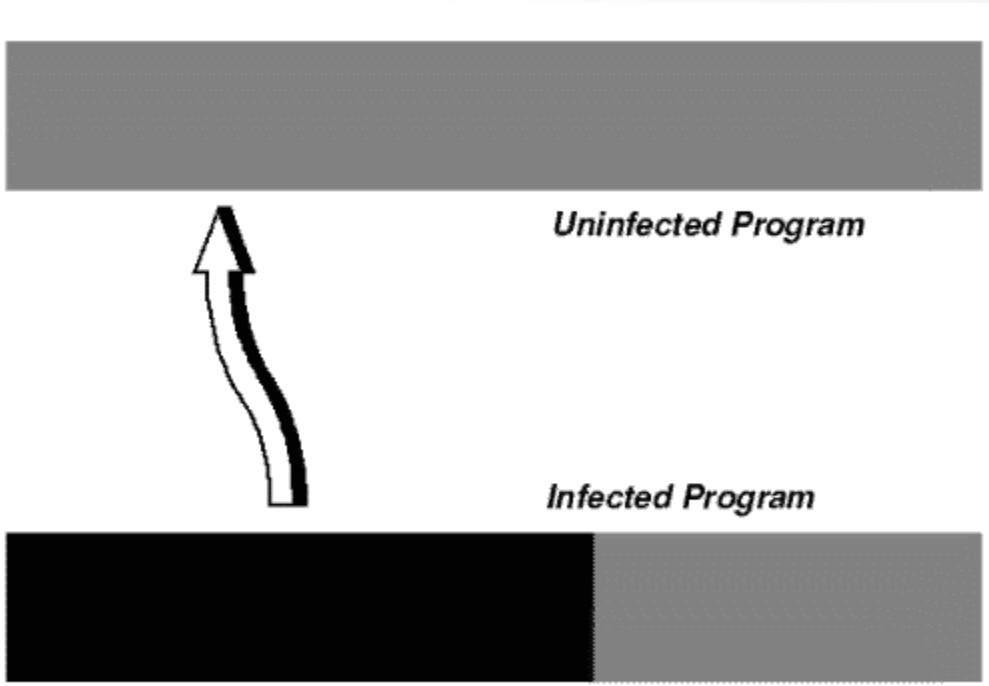
- little to do with Unix shell
- "wrapping around" a given program or system call
  - the original program can be even copied and stored elsewhere = Companion Virus
    - example: p.com and p.exe
- the infected program becomes a subroutine of the virus code
- controls, hijacks and isolates the given program/routine completely



Intrusive Virus

Does not append,
rather modifies the program itself and changes the functionality of this program.

Cannot be removed if we don't have the original copy…



Viruses by Medium of Infection

- Exe file infectors => most viruses
- Boot infectors
  - hard drive boot
    - master boot record (MBR)
    - OS loader hijack
    - UEFI infection
  - CDROM autorun hijackers
  - USB stick autorun hijackers
- Half way before system starts:
  - OS libraries hijack (e.g. some dll loaded early)
  - driver hijackers
- Data file infectors
  - Macro Viruses
  - format string exploits (not called viruses)

Viruses by Medium of Infection

- Exe file infectors
- Boot infectors
  - hard drive boot
    - master boot record (MBR), OS indep
    - OS loader hijack, 1 partition
  - CDROM autorun hijackers
  - USB stick autorun hijackers
- Half way before system starts:
  - OS libraries hijack (e.g. some dll loaded early)
  - driver hijackers
- Data file infectors
  - Macro Viruses
  - format string exploits (not called viruses)

load before any anti-virus software

Additional Infection Mechanisms

- Terminate and Stay Resident = TSR
  - since MS-DOS..
  - stays active in memory after application exits
  - can then infect other targets, for example
    - can trap OS calls that execute any program…

Virus Defenses

Oldest methods:

- Black-list:
  - signature-based detection.
- Track changes to executables:
  - Tripwire, hash functions, MACs etc…

Virus Self-Defense

"Stealth" Viruses – avoid detection

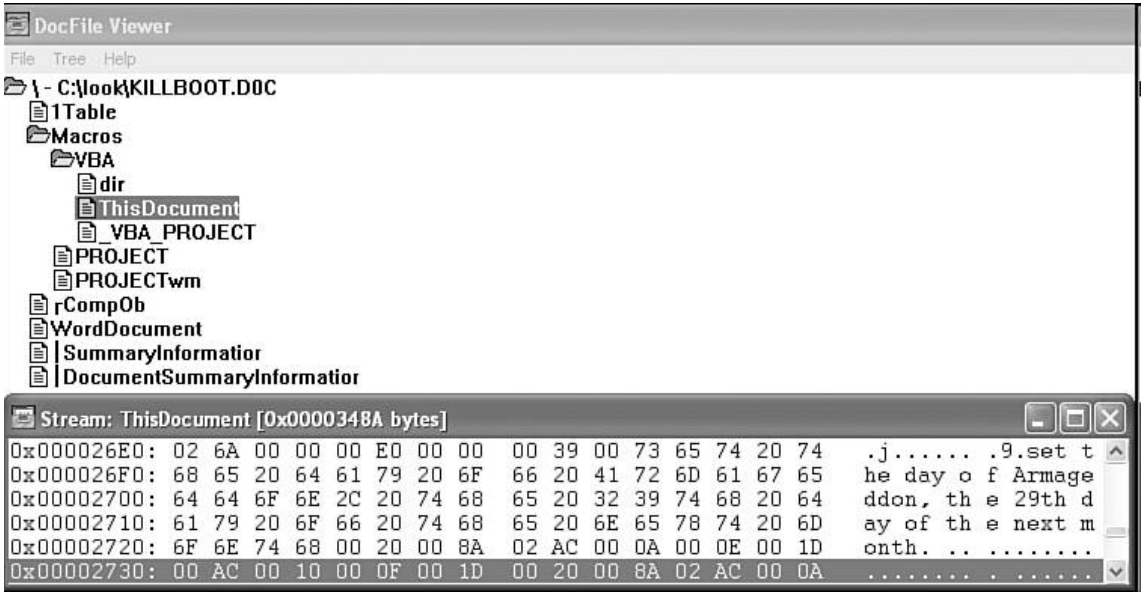
- conceal code:
  - Pack/compress/encrypt virus
  - Polymorphism
    - constantly change virus code
- conceal actions
  - mimicry: imitate other programs
  - associated rootkit prevents detection
  - watchdog program
  - disable or disturb anti-virus software
  - remove itself after job done, such as creating 2 copies elsewhere

Self-defence technology	Relative frequency
Packers	47.8%
Code obfuscation	14.3%
Rootkits	14.3%
Disrupt security software	9.6%
Virtual machine detection	4.8%
Auto-click	4.8%
Anti-debugging	1.9%
Polymorphism	1.0%
Block file access	1.0%
Modify hosts file	0.5%
Using streams	0.1%

Table 6. Relative prevalence of malware self-defence technolgies identified by Shevchenko [13].

Macro Viruses

- infected a data file (e.g. word)
  - relies on macros interpreted by some application
    - application-dependent
    - can be OS-independent
      - Example: Microsoft Word: MAC and Windows



Independent / “More Sophisticated Forms of Life”

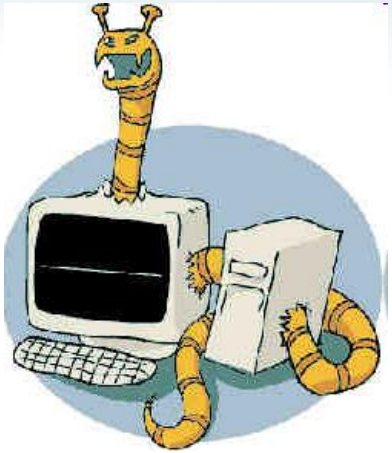
*Bacteria

Bacteria:

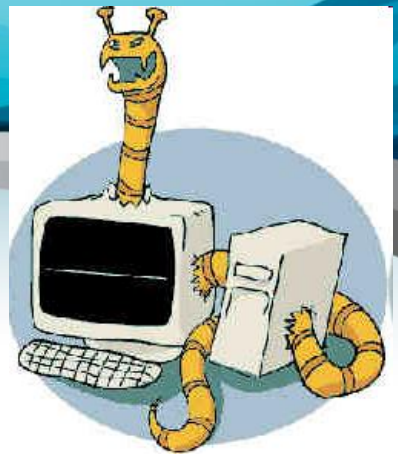
simple functionality,
program that replicates until it fills all disk space, all memory, all CPU cycles

Worms

- introduced by Shoch and Hupp in 1982.
- runs independently,
  - no host program
  - infects a host machine
  - propagates in a network
    - a fully working version of itself copied to another host machine
    - spreads totally without human intervention ≠ virus



more worms



A worm has two main components:

- an exploit
  - usually exploits web servers
    - or other exposed "DMZ-style" components
- a "payload" of hidden tasks
  - backdoors, spam relays, DDoS agents, etc.

Life cycle phases:

probing ➔ exploitation ➔ replication ➔ running the payload

Zombie Network = Botnet

- Secretly takes over another networked computer by exploiting software flaws
- Connect the compromised computers into a zombie network or botnet =
  - a collection of compromised machines
    - running programs such as worms, Trojan horses, or backdoors,
    - under a common command and control infrastructure.
- Uses it to indirectly launch attacks
  - e.g., spamming, phishing , DDoS, password cracking etc.
- very frequently sold or rented,
  - about 0.05 $ / host / week

Rootkits

- Software used after system compromise to:
  - Hide the attacker's presence
  - Provide backdoors for easy reentry

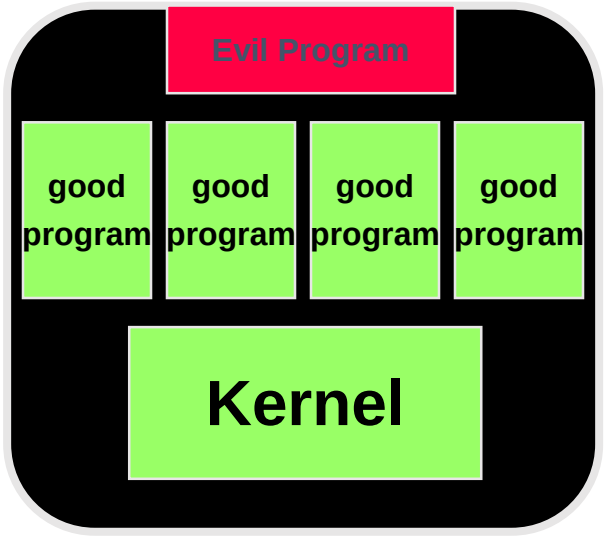
- Simple rootkits:
  - Modify user programs (ls, ps)
  - Modify a compiler
  - Detectable by tools like Tripwire (stores hashes of files).

- Sophisticated rootkits:

  - Modify the kernel itself
  - Hard to detect from userland

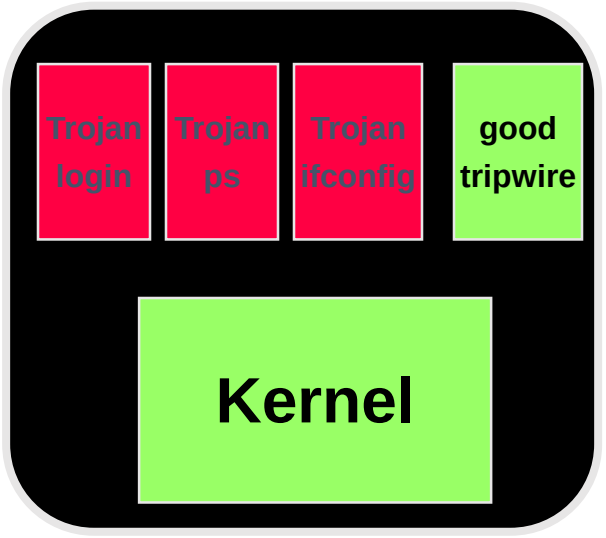
Rootkit Classification (1)

Application-level Rootkit



Hxdef, NTIllusion

Traditional RootKit

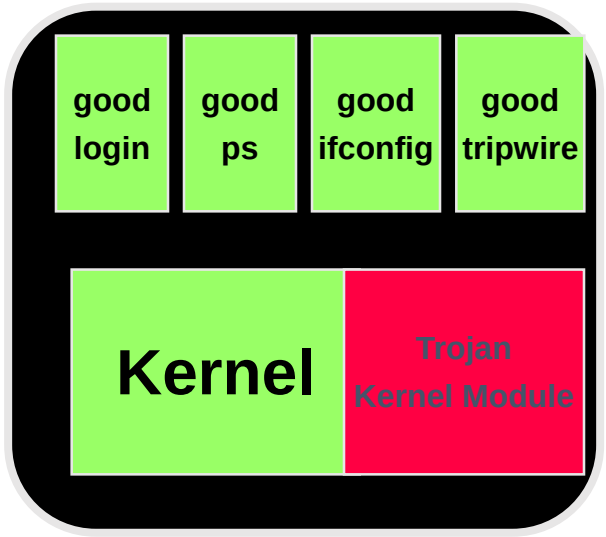


Lrk5, t0rn

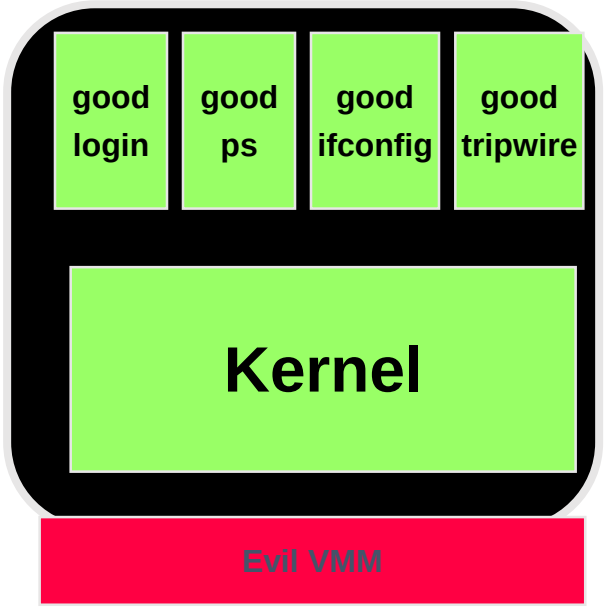
Tripwire: detected! - maybe not detected

Rootkit Classification (2)

Kernel-level RootKit



Under-Kernel RootKit



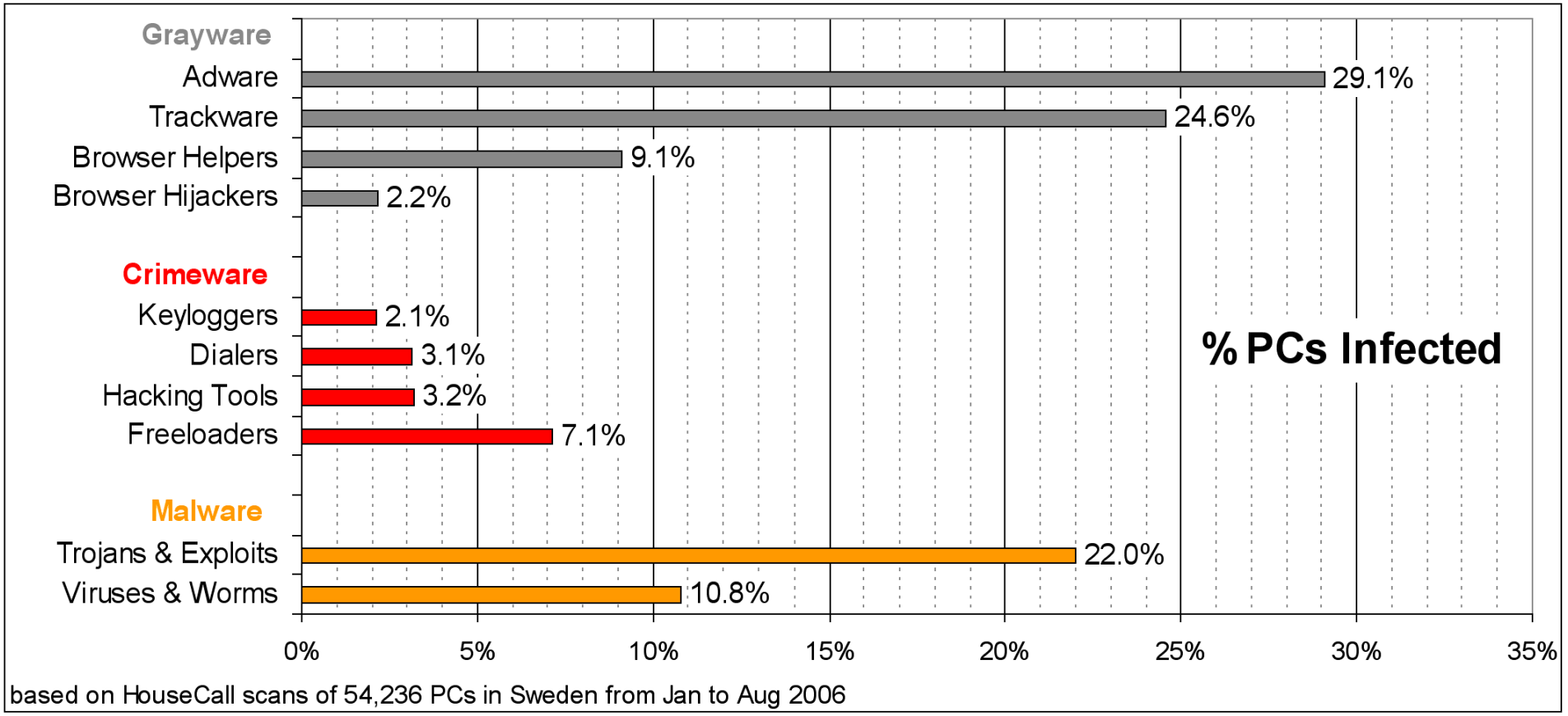
Shadow Walker, adore

SubVirt, ``Blue Pill''

What's Going On?

Is My PC Infected?

- Swedish large scale study



Another Study

PCs infected
[source: Trend Micro]

1. Adware	43.9%
2. Trojan	18.8%
3. Browser helper	13.8%
4. Freeloader	9.8%
5. Trojan spyware	9.6%
6. Trackware	9.3%
7. Cracking app	5.0%
8. Worm	4.3%
9. Java script	3.9%
10. Dialler	3.1%
11. Keylogger	2.9%
12. Hacking tool	2.7%
13. Backdoor	2.1%
14. Portable executable	1.9%
15. Downloader	1.8%
16. Remote access app	1.2%
17. Exploit	1.1%
18. HTML script	0.7%
19. Joke program	0.6%
20. Browser hijacker	0.4%

According to experts:

- Today's malware is designed to remain undetected for months.
  - do not get famous, get rich!
  - zero-day malware

Questions