Cryptography and Network Security

Chapter 11

Malicious Software

Phần mềm độc hại

Lectured by
Nguyễn Đức Thái

Outline

- Types of Malicious Software các loại phần mềm độc hại
- Viruses
- Virus Countermeasures Biện pháp đối phó Virus
- Worms
- Distributed Denial of Service Attacks

Key Points

Phần mềm độc hại là phần mềm được cố tình hoặc chèn vào trong một hệ thống cho một mục đích gây hại

- ***Malicious software*** is software that is ***intentionally*** included or inserted in a system for a ***harmful purpose***.
- A ***virus*** is a piece of software that can "infect" other programs by modifying them; the modification includes a copy of the virus program, which can then go on to infect other programs.

lây nhiễm

Một virus là một phần của phần mềm mà có thể "lây nhiễm" các chương trình khác bằng cách sửa đổi chúng; việc sửa đổi bao gồm một bản sao của các chương trình virus, sau đó có thể đi để lây nhiễm vào các chương trình khác.

- A ***worm*** is a program that can ***replicate*** itself and ***send copies*** from computer to computer across network connections.

Một con sâu là một chương trình có thể tự tái tạo và gửi bản sao từ máy tính đến máy tính thông qua kết nối mạng.

  - Upon arrival, the worm may be activated to replicate and propagate again.
  - In addition to propagation, the worm usually performs some unwanted function.

Khi đến nơi, sâu có thể được kích hoạt để tái tạo và tuyên truyền lần nữa.

Ngoài nhân giống, worm thường thực hiện một số chức năng không mong muốn.

Key Points

- A ***denial of service*** (DoS) attack is an attempt to prevent legitimate users of a service from using that service.
- A ***distributed denial of service (DDOS)*** attack is launched ***from multiple*** coordinated ***sources***.

Một sự từ chối dịch vụ (DoS) tấn công là một nỗ lực để ngăn chặn người dùng hợp pháp một dịch vụ từ việc sử dụng dịch vụ đó

Một tấn công từ chối dịch vụ tấn công (DDOS) được khởi động từ nhiều nguồn phối hợp

Intro

- Perhaps the ***most sophisticated types of threats*** to computer systems are presented by programs that exploit ***vulnerabilities*** in computing systems.
- Such threats are referred to as ***malicious software***, or ***malware***.
- In this context, we are concerned with threats to ***application programs*** as well as ***utility programs***, such as editors and compilers, and kernel-level programs.

Có lẽ các loại phức tạp nhất của các mối đe dọa hệ thống máy tính được đại diện bởi các chương trình khai thác lỗ hổng trong hệ thống máy tính

các mối đe dọa như vậy được gọi là các phần mềm có hại, hoặc phần mềm độc hại

chúng ta đang quan tâm đến các mối đe dọa cho các chương trình ứng dụng cũng như các chương trình tiện ích, chẳng hạn như biên tập và trình biên dịch, và các chương trình hạt nhân cấp.

Types of Malicious Software

Phần mềm độc hại có thể được chia thành hai loại

- Malicious software can be divided into ***two categories***:
  - those that need a host program, and
  - those that are independent.

- The former, referred to as ***parasitic***, are essentially fragments of programs that cannot exist independently of some actual application program, utility, or system program. **Viruses**, **logic bombs** and **backdoors** are examples
- Independent malware is a self-contained program that can be scheduled and run by the operating system. **Worms** and **bot programs** are examples

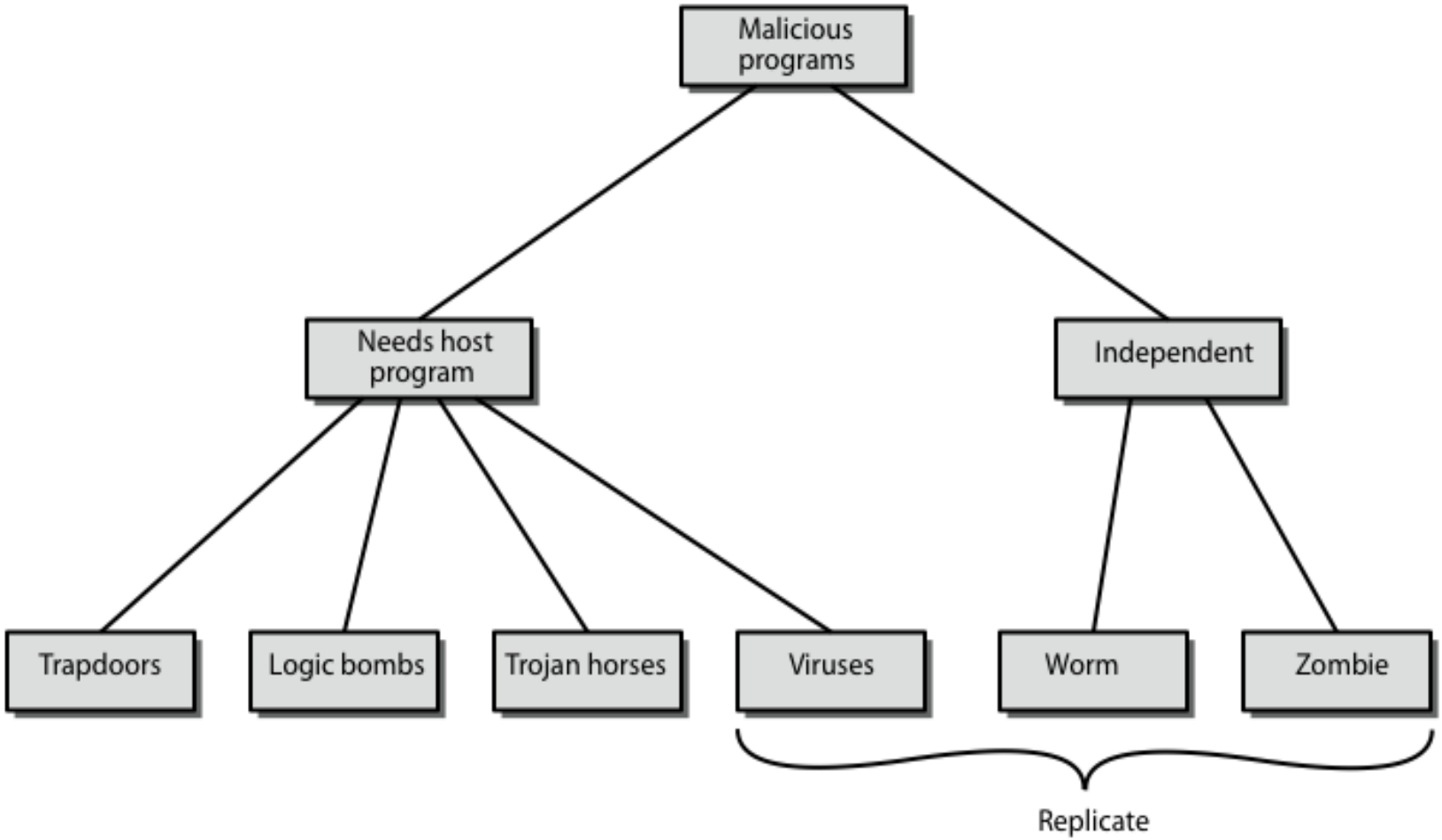
Types of Malicious Software

- We can also differentiate between those software threats that ***do not replicate*** and ***those that do***.

- The former are ***programs*** or ***fragments of programs*** that are activated by a trigger. Examples are logic bombs, backdoors, and bot programs.

- The latter consist of **either** a ***program fragment*** **or** an ***independent program*** that, when executed, may produce one or more copies of itself to be activated later on the same system or some other system. Viruses and worms are examples.

Types of Malicious Software



Backdoor

- A ***backdoor***, also known as a ***trapdoor***, is a ***secret entry point*** into a program that allows someone who is aware of the backdoor to gain access without going through the usual security access procedures.
- Programmers have used backdoors legitimately for many years to debug and test programs; such a backdoor is called a ***maintenance hook***
- This usually is done when the programmer is developing an application that has an authentication procedure, or a long setup, requiring the user to enter many different values to run the application. To debug the program, the developer may wish to gain special privileges or to avoid all the necessary setup and authentication

Backdoor

- The programmer may also want to ensure that there is a method of activating the program should something be wrong with the authentication procedure that is being built into the application.
- The ***backdoor*** is ***code*** that recognizes some special sequence of input or is triggered by being run from a certain user ID or by an unlikely sequence of events.
- Backdoors become ***threats*** when unscrupulous programmers use them to gain unauthorized access
- It is ***difficult*** to implement operating system controls for backdoors.
- Security measures must ***focus on*** the ***program development*** and ***software update activities***.

Logic bomb

- ***One of the oldest types of program threat***, predating viruses and worms, is the ***logic bomb***.
- The logic bomb is ***code*** embedded in some legitimate program that is set to "explode" when certain conditions are met.
- Examples of conditions that can be used as triggers for a logic bomb are the presence or absence of certain files, a particular day of the week or date, or a particular user running the application.
- Once triggered, a bomb may ***alter*** or ***delete*** data or entire files, cause a machine halt, or do some other damage

Trojan horses

- A Trojan horse is a useful, or apparently useful, ***program*** or ***command procedure*** containing hidden code that, when invoked, performs some unwanted or harmful function.
- Trojan horse programs can be used to accomplish functions ***indirectly*** that an unauthorized user could not accomplish directly
  - For example, to gain access to the files of another user on a shared system, a user could create a Trojan horse program that, when executed, changes the invoking user's file permissions so that the files are readable by any user.
  - The author could then induce users to run the program by placing it in a common directory and naming it such that it appears to be a useful utility program or application

Trojan horses

Trojan horses fit into one of three models:

- Continuing to perform the function of the original program and ***additionally*** performing a separate malicious activity
- Continuing to perform the function of the original program but ***modifying the function to perform malicious activity*** (e.g., a Trojan horse version of a login program that collects passwords) or to disguise other malicious activity (e.g., a Trojan horse version of a process listing program that does not display certain processes that are malicious)
- Performing a malicious function that completely replaces the function of the original program

Mobile code

- Mobile code refers to ***programs*** (e.g., script, macro, or other portable instruction) that can be shipped unchanged to a heterogeneous collection of platforms and ***execute with identical semantics*** .
- Mobile code is transmitted ***from a remote system to a local system*** and then ***executed on the local system*** without the user's explicit instruction.
- Mobile code often acts as a mechanism for a virus, worm, or Trojan horse to be transmitted to the user's workstation.
- In other cases, mobile code takes advantage of vulnerabilities to perform its own exploits, such as unauthorized data access or root compromise

Multiple-Threat Malware

- Viruses and other malware may operate in ***multiple ways***.
- A multipartite virus infects in multiple ways.
- Typically, the ***multipartite*** virus is capable of infecting ***multiple types of files***, so that virus eradication must deal with all of the possible sites of infection
- A blended attack uses ***multiple methods*** of infection or transmission, to maximize the speed of contagion and the severity of the attack.

Multiple-Threat Malware

- An example of a blended attack is the Nimda attack, erroneously referred to as simply a worm.
- Nimda uses four distribution methods:
  - **E-mail**: A user on a vulnerable host opens an infected e-mail attachment; Nimda looks for e-mail addresses on the host and then sends copies of itself to those addresses.
  - **Windows shares**: Nimda scans hosts for unsecured Windows file shares; it can then use NetBIOS86 as a transport mechanism to infect files on that host in the hopes that a user will run an infected file, which will activate Nimda on that host.
  - **Web servers**: Nimda scans Web servers, looking for known vulnerabilities in Microsoft IIS. If it finds a vulnerable server, it attempts to transfer a copy of itself to the server and infect it and its files.
  - **Web clients**: If a vulnerable Web client visits a Web server that has been infected by Nimda, the client's workstation will become infected.

Viruses

- piece of software that infects programs
  - modifying them to include a copy of the virus
  - so it executes secretly when host program is run
- specific to operating system and hardware
  - taking advantage of their details and weaknesses

Virus structure

A computer virus has three parts:

- ***Infection mechanism***: The ***means*** by which a virus spreads, enabling it to replicate. The mechanism is also referred to as the infection vector.
- ***Trigger***: The event or condition that determines when the payload is activated or delivered.
- ***Payload***: What the virus does, besides spreading. The payload may involve damage or may involve benign but noticeable activity.

Virus structure

```
    program V :=

{goto main;
    1234567;

    subroutine infect-executable :=
        {loop:
        file := get-random-executable-file;
        if (first-line-of-file = 1234567)
          then goto loop
          else prepend V to file; }

    subroutine do-damage :=
        {whatever damage is to be done}

    subroutine trigger-pulled :=
        {return true if some condition holds}

main:   main-program :=
        {infect-executable;
        if trigger-pulled then do-damage;
        goto next;}

next:

}
```

Virus phases

A typical virus goes through the following four phases:

- ***Dormant phase***: The virus is **idle**. The virus will eventually be activated by some event, such as a date, the presence of another program or file, or the capacity of the disk exceeding some limit. Not all viruses have this stage.
- ***Propagation phase***: The virus places a copy of itself into other programs or into certain system areas on the disk. The copy may not be identical to the propagating version; viruses often morph to evade detection. Each infected program will now contain a clone of the virus, which will itself enter a propagation phase.
- ***Triggering phase***: The virus is **activated** to perform the function for which it was intended. As with the dormant phase, the triggering phase can be caused by a variety of system events, including a count of the number of times that this copy of the virus has made copies of itself
- ***Execution phase***: The function is performed. The function may be harmless, such as a message on the screen, or damaging, such as the destruction of programs and data files

Virus classifications

A virus classification ***by target*** includes the following categories:

- ***Boot sector infector***: Infects a master boot record or boot record and spreads when a system is booted from the disk containing the virus.
- ***File infector***: Infects files that the operating system or shell consider to be executable.
- ***Macro virus***: Infects files with macro code that is interpreted by an application.

Virus classifications

A virus classification by ***concealment strategy*** includes the following categories:

- ***Encrypted virus***: A *portion of the virus* creates a *random encryption key* and encrypts the remainder of the virus. The key is stored with 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. Because the bulk of the virus is encrypted with a different key for each instance, there is no constant bit pattern to observe.
- ***Stealth virus***: A form of virus explicitly designed ***to hide itself from detection*** by antivirus software. Thus, the entire virus, not just a payload is hidden.
- ***Polymorphic virus***: A virus that mutates with every infection, making detection by the "signature" of the virus impossible.
- ***Metamorphic virus***: As with a polymorphic virus, a metamorphic virus mutates with every infection. The difference is that a metamorphic virus rewrites itself completely at each iteration, increasing the difficulty of detection.

Macro viruses

- In the mid-1990s, macro viruses became by far the most prevalent type of virus.
- Macro viruses are particularly threatening for a number of reasons:
  1. ***A macro virus is platform independent***. Many macro viruses infect Microsoft Word documents or other Microsoft Office documents. Any hardware platform and operating system that supports these applications can be infected.
  2. ***Macro viruses infect documents***, not executable portions of code. Most of the information introduced onto a computer system is in the form of a document rather than a program.
  3. ***Macro viruses are easily spread***. A very common method is by electronic mail.
  4. Because macro viruses infect user documents rather than system programs, traditional file system access controls are of limited use in preventing their spread

E-mail Viruses

- A more recent development in malicious software is the ***e-mail virus***.
- The first rapidly spreading e-mail viruses, such as ***Melissa***, made use of a Microsoft Word macro embedded in an attachment.
- If the recipient opens the e-mail attachment, the Word macro is activated. Then
  1. The e-mail virus sends ***itself*** to everyone on the mailing list in the user's e-mail package.
  2. The virus does local damage on the user's system.

Virus Countermeasures

- ***prevention*** **- ideal solution but difficult**
- **realistically need:**
  - detection
  - identification
  - removal
- **if detected but can't identify or remove, then the *alternative* is to discard the infected file and reload a clean backup version.**

Anti-Virus Evolution

- ***Advances*** in virus and antivirus technology go hand in hand. Early viruses were relatively ***simple code*** fragments and could be identified and purged with relatively ***simple antivirus software*** packages.
- As the virus arms race has evolved, ***both viruses*** and, necessarily, ***antivirus*** software have grown more complex and sophisticated.
- Generations
  - ***First generation***: simple scanners
  - ***Second generation***: heuristic scanners
  - ***Third generation***: activity traps
  - ***Fourth generation***: full-featured protection

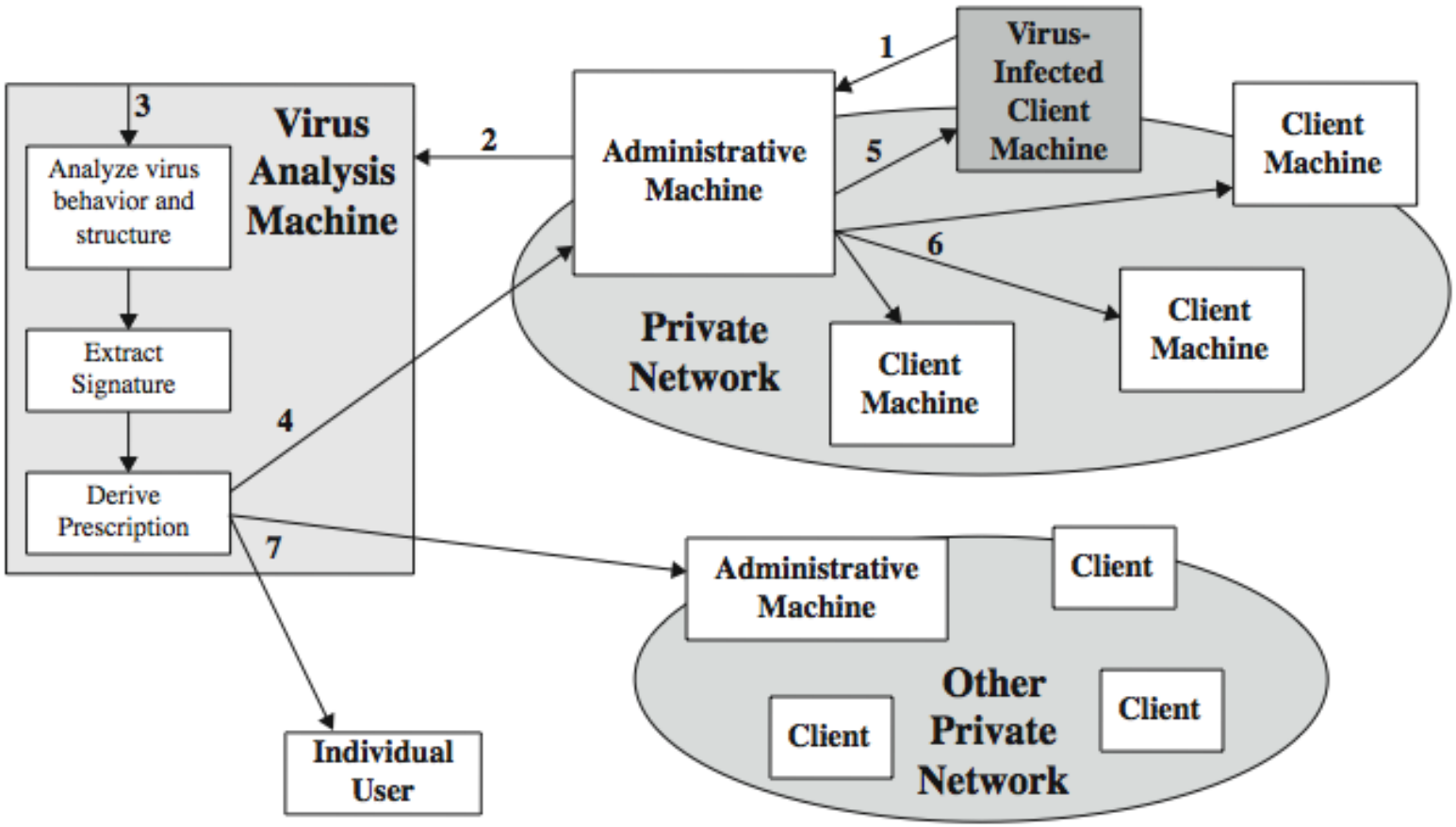
Advanced Antivirus Techniques

- Generic Decryption
- Digital Immune System
- Behavior-blocking Software

Generic Decryption (GD)

- ***Generic decryption*** (GD) technology enables the antivirus program to ***easily detect*** even the most complex polymorphic viruses while maintaining fast scanning speeds
- When a file containing a polymorphic virus is executed, the virus must ***decrypt itself*** to activate. In order to detect such a structure, executable files are run through a GD scanner, which contains the following elements:
  - ***CPU emulator:*** A software-based virtual computer. Instructions in an executable file are interpreted by the emulator rather than executed on the underlying processor
  - ***Virus signature scanner:*** to check known virus signatures
  - ***Emulation control module:*** to manage process

Digital Immune System



Digital Immune System

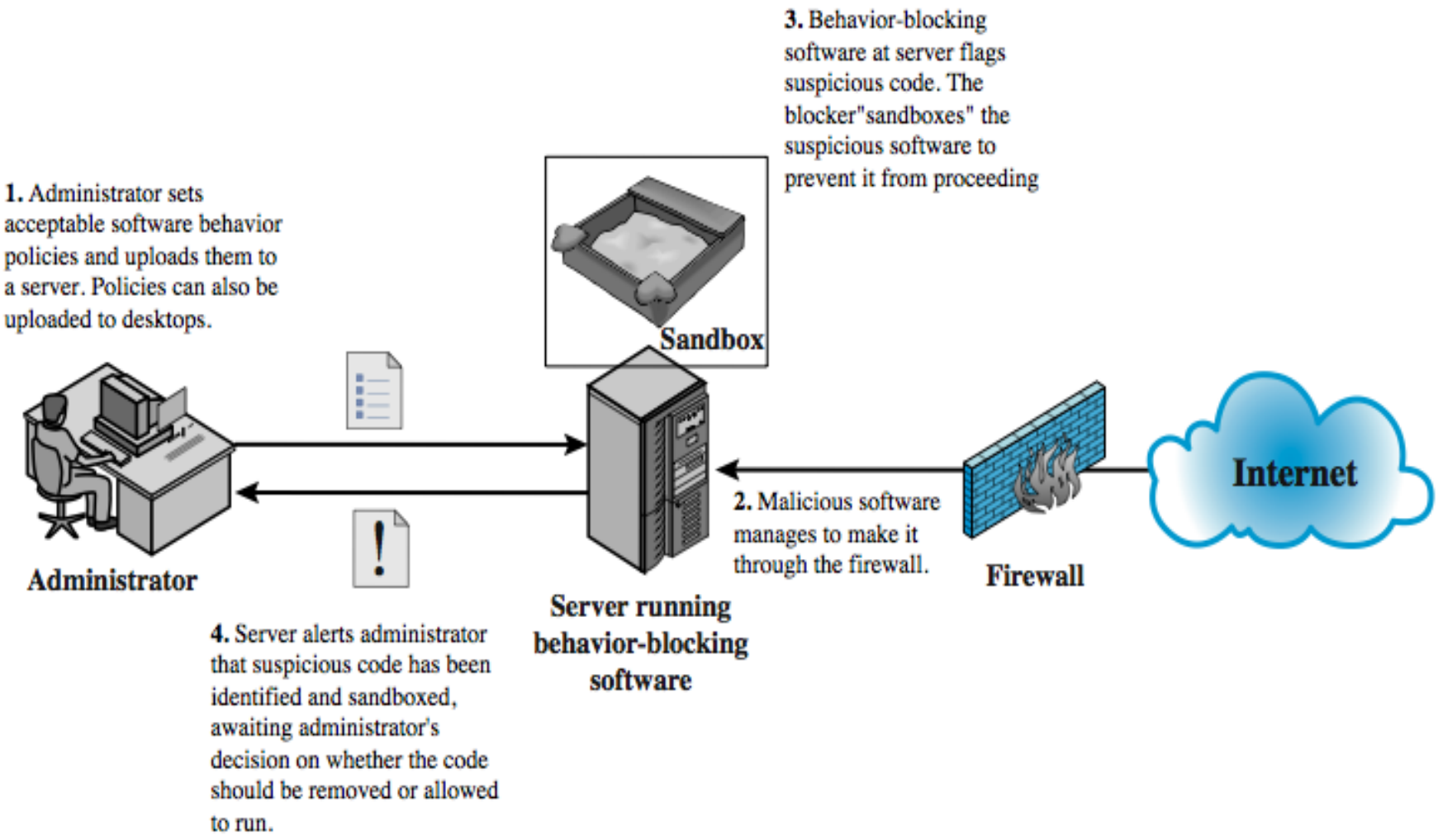
Typical steps in digital immune system operation (illustrated in the figure):

1. A monitoring program on each PC uses a variety of heuristics based on system behavior, suspicious changes to programs, or family signature to infer that a virus may be present.The monitoring program forwards a copy of any program thought to be infected to an administrative machine within the organization.
2. The *administrative machine* encrypts the sample and sends it to a central virus analysis machine.
3. This machine creates an environment in which the infected program can be safely run for analysis.Techniques used for this purpose include emulation, or the creation of a protected environment within which the suspect program can be executed and monitored. The virus analysis machine then produces a prescription for identifying and removing the virus.
4. The resulting prescription is sent back to the administrative machine.
5. The administrative machine forwards the prescription to the infected client.
6. The prescription is also forwarded to other clients in the organization.
7. Subscribers around the world receive regular antivirus updates that protect them from the new virus.

Behavior-Blocking Software

- Unlike heuristics or fingerprint-based scanners, behavior-blocking software ***integrates with the operating system*** of a host computer and monitors program behavior in real-time for malicious .
- The behavior blocking software then blocks potentially malicious actions before they have a chance to affect the system. Monitored behaviors can include
  - Attempts to open, view, delete, and/or modify files;
  - Attempts to format disk drives and other unrecoverable disk operations;
  - Modifications to the logic of executable files or macros;
  - Modification of critical system settings, such as start-up settings;
  - Scripting of e-mail and instant messaging clients to send executable content; and
  - Initiation of network communications.

Behavior-Blocking Software



Worms (1/5)

- A worm is a ***program*** that can replicate itself and send copies from computer to computer across network connections.
- Upon arrival, the worm may be ***activated*** to replicate and ***propagate*** again.
- In addition to propagation, the worm usually performs some unwanted function.
- An ***e-mail virus*** has some of the characteristics of a worm because it propagates itself from system to system.
- However, ***we can still classify it as a virus*** because it uses a document modified to contain viral macro content and requires human action.
- A worm actively ***seeks out more machines*** to infect and each machine that is infected serves as an automated launching pad for attacks on other machines.

Worms (2/5)

- Network worm programs ***use network connections*** to spread from system to system.
- Once active ***within a system***, a network worm can ***behave as a*** computer ***virus*** or ***bacteria***, or it could implant Trojan horse programs or perform any number of disruptive or destructive actions

Worms (3/5)

- To ***replicate itself***, a network worm uses some sort of network vehicle:
  - **Electronic mail facility**: A worm mails a copy of itself to other systems, so that its code is run when the e-mail or an attachment is received or viewed.
  - **Remote execution capability**: A worm executes a copy of itself on another system, either using an explicit remote execution facility or by exploiting a program flaw in a network service to subvert its operations.
  - **Remote login capability**: A worm logs onto a remote system as a user and then uses commands to copy itself from one system to the other, where it then executes.

Worms (4/5)

- The new copy of the worm program is then ***run on the remote system*** where, in addition to any functions that it performs at that system, ***it continues to spread in the same fashion***.
- A network worm exhibits the ***same characteristics*** as a computer virus: ***a dormant phase***, ***a propagation phase***, ***a triggering phase***, and ***an execution phase***.
- The propagation phase generally performs the following functions:
  1. Search for other systems to infect by examining host tables or similar repositories of remote system addresses.
  2. Establish a connection with a remote system.
  3. Copy itself to the remote system and cause the copy to be run.

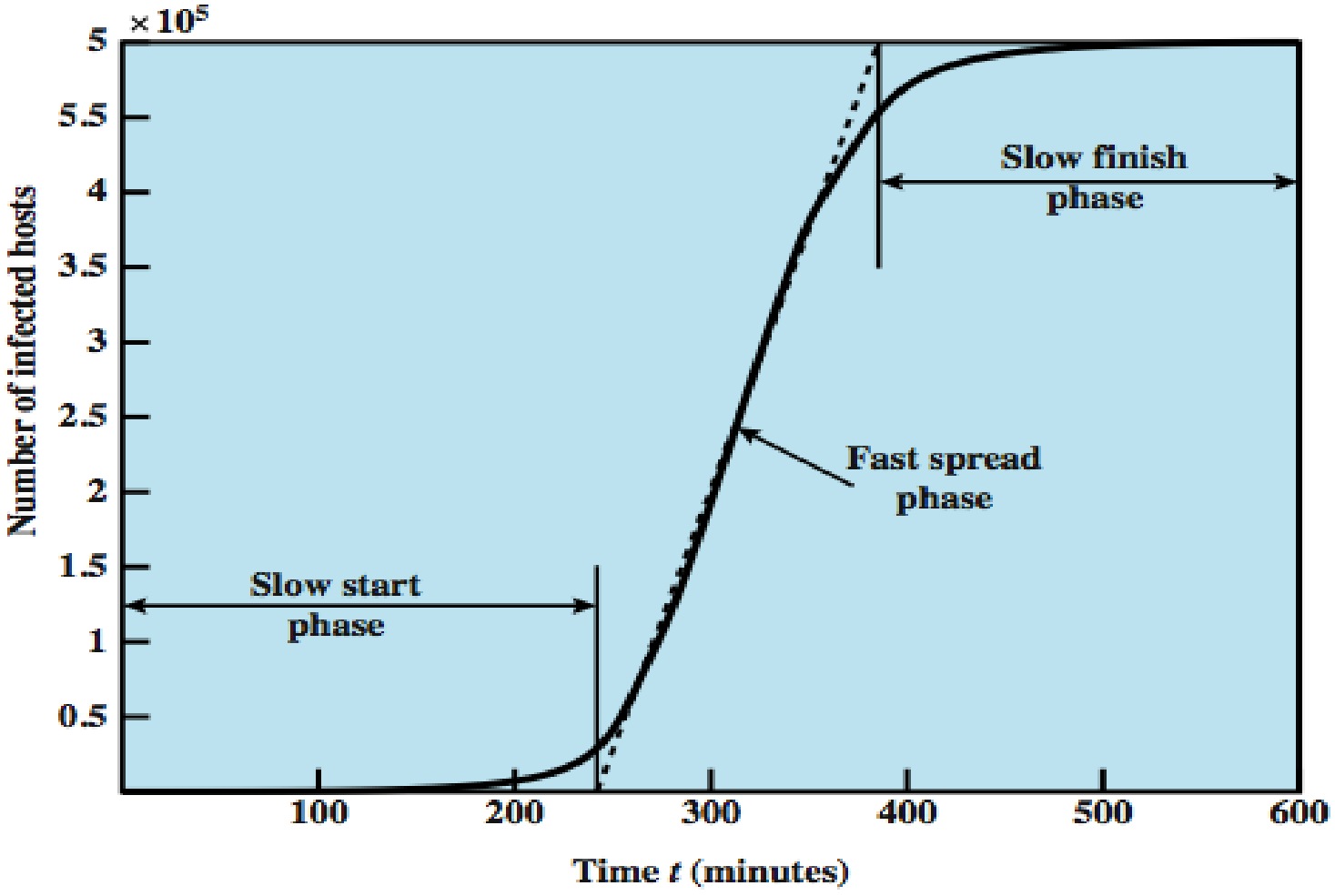
Worms (5/5)

- The network worm may also ***attempt to determine*** whether a system has ***previously*** been ***infected*** before copying itself to the system.
- In a multiprogramming system, it may also disguise its presence by naming itself as a system process or using some other name that may not be noticed by a system operator.
- As with viruses, network worms are difficult to counter.

The Morris Worms

- **one of best know worms**
- **released by Robert Morris in 1988**
- **various attacks on UNIX systems**
  - cracking password file to use login/password to logon to other systems
  - exploiting a bug in the finger protocol
  - exploiting a bug in sendmail
- **if succeed, have remote shell access**
  - sent bootstrap program to copy worm over

Worm Propagation Model



Recent Worm Attacks

- **Code Red**
  - July 2001 exploiting MS IIS bug
  - probes random IP address, does DDoS attack
- **Code Red II variant includes backdoor**
- **SQL Slammer**
  - early 2003, attacks MS SQL Server
- **Mydoom**
  - mass-mailing e-mail worm that appeared in 2004
  - installed remote access backdoor in infected systems
- **Warezov family of worms**
  - scan for e-mail addresses, send in attachment

Worm Technology (1/2)

- ***Multiplatform***: Newer worms are ***not limited*** to Windows machines but can attack a variety of platforms, especially the popular varieties of UNIX.
- ***Multi-exploit***: New worms penetrate systems in a variety of ways, using exploits against Web servers, browsers, e-mail, file sharing, and other network-based applications.
- ***Ultrafast spreading***: One technique to accelerate the spread of a worm is to conduct a prior Internet scan to accumulate Internet addresses of vulnerable machines.
- ***Polymorphic***: To evade detection, skip past filters, and foil real-time analysis, worms adopt the virus polymorphic technique. Each copy of the worm has new code generated on the fly using functionally equivalent instructions and encryption techniques.

Worm Technology (2/2)

- ***Metamorphic***: In addition to changing their appearance, metamorphic worms have a repertoire of behavior patterns that are unleashed at different stages of propagation.
- ***Transport vehicles***: Because worms can rapidly compromise a large number of systems, they are ideal for spreading other distributed attack tools, such as distributed denial of service bots.
- ***Zero-day exploit***: To achieve maximum surprise and distribution, a worm should exploit an unknown vulnerability that is only discovered by the general network community when the worm is launched.

Mobile Phone Worms

- Worms first appeared on mobile phones in 2004.
- These worms communicate through Bluetooth wireless connections or via the multimedia messaging service
- ***The target is the smartphone***, which is a mobile phone that permits users to install software applications from sources other than the cellular network operator.
- Mobile phone malware can completely disable the phone, delete data on the phone, or force the device to send costly messages to premium-priced numbers.

Mobile Phone Worms - Example

- An example of a mobile phone worm is ***CommWarrior***, which was launched in 2005.
- This worm replicates by means of ***Bluetooth*** to other phones in the receiving area.
- It also sends itself as an MMS file to numbers in the phone's address book and in automatic replies to incoming text messages and MMS messages.
- In addition, it copies itself to the removable memory card and inserts itself into the program installation files on the phone.

Worms Countermeasures (1/2)

- There is considerable overlap in techniques for dealing with viruses and worms.
- Once a worm is resident on a machine, ***antivirus software can be used to detect it***.
- In addition, because worm propagation generates considerable network activity, network activity and usage monitoring can form the basis of a worm defense.
- To begin, let us consider the requirements for an effective worm countermeasure scheme:
  1. **Generality**: The approach taken should be able to handle a wide variety of worm attacks, including polymorphic worms.
  2. **Timeliness**: The approach should respond quickly so as to limit the number of infected systems and the number of generated transmissions from infected systems.

Worms Countermeasures (2/2)

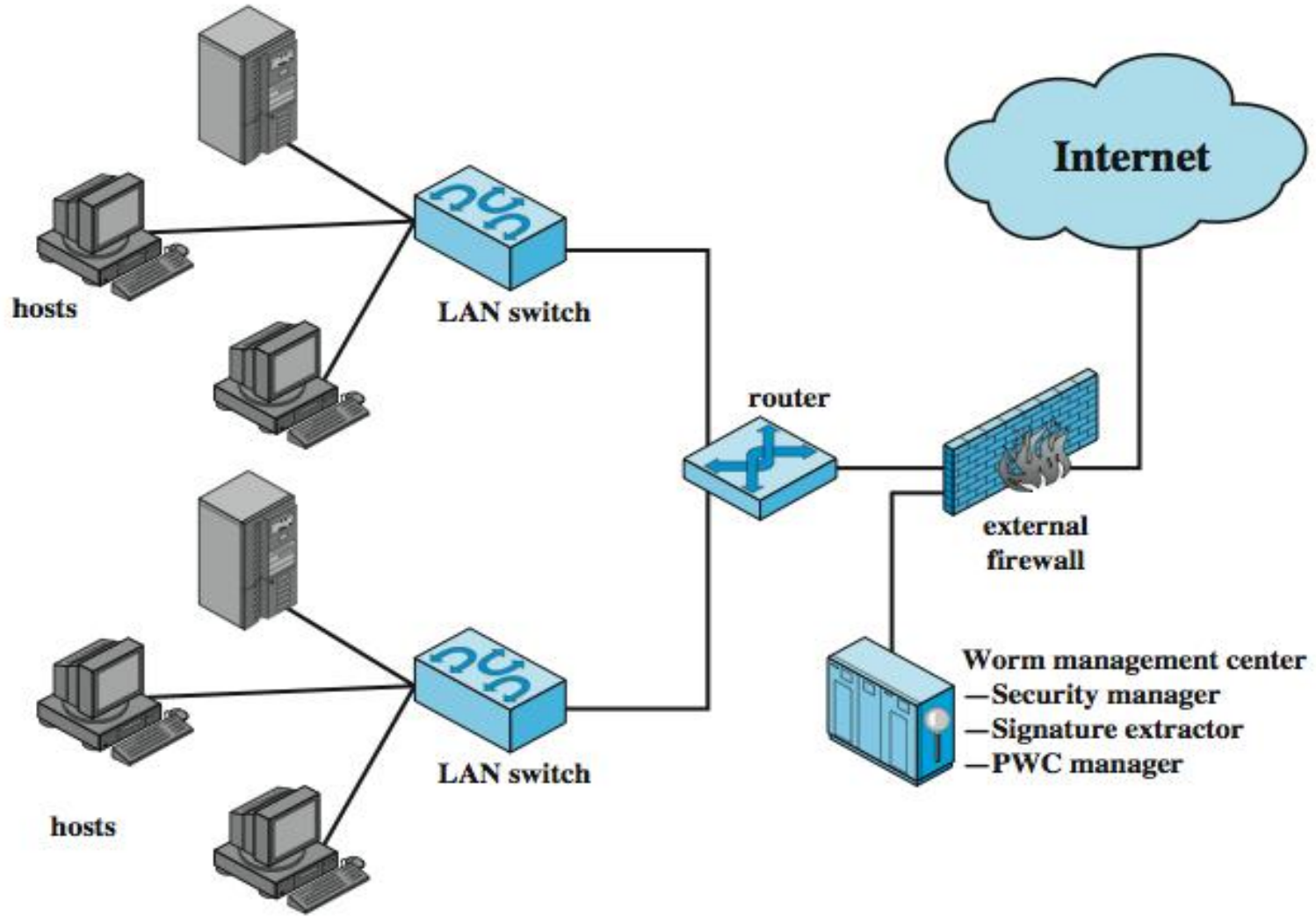
3. **Resiliency**: The approach should be resistant to evasion techniques employed by attackers to evade worm countermeasures.
4. **Minimal denial-of-service costs**: The approach should result in minimal reduction in capacity or service due to the actions of the countermeasure software. That is, in an attempt to contain worm propagation, the countermeasure should not significantly disrupt normal operation.
5. **Transparency**: The countermeasure software and devices should not require modification to existing (legacy) OSs, application software, and hardware.
6. **Global and local coverage**: The approach should be able to deal with attack sources both from outside and inside the enterprise network.

Worms - conclusion

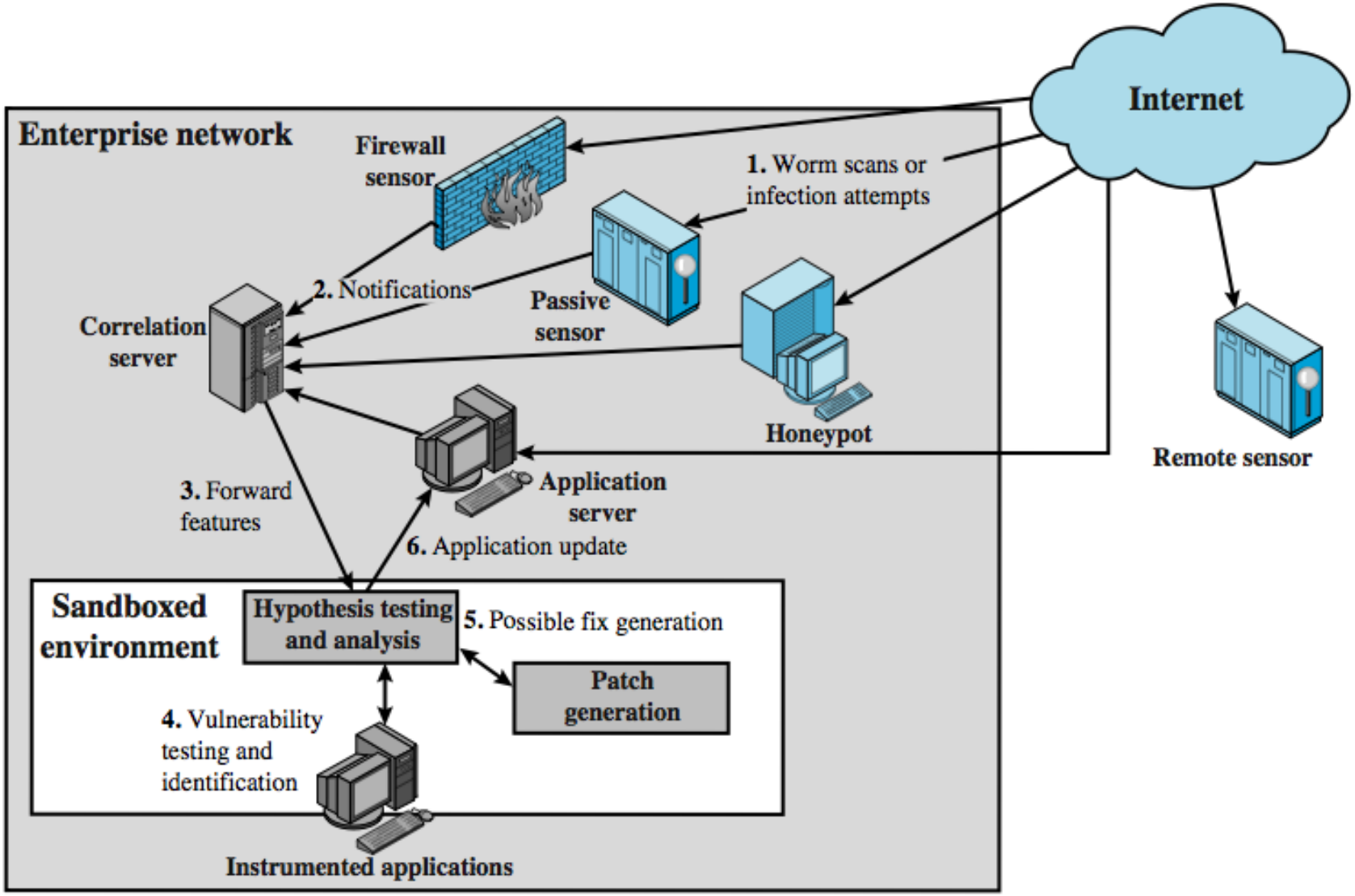
No existing worm ***countermeasure scheme*** *appears to satisfy all these requirements.*

➔ *Thus, administrators typically need to use **multiple approaches** in defending against worm attacks.*

Pro-Active Worms Containment



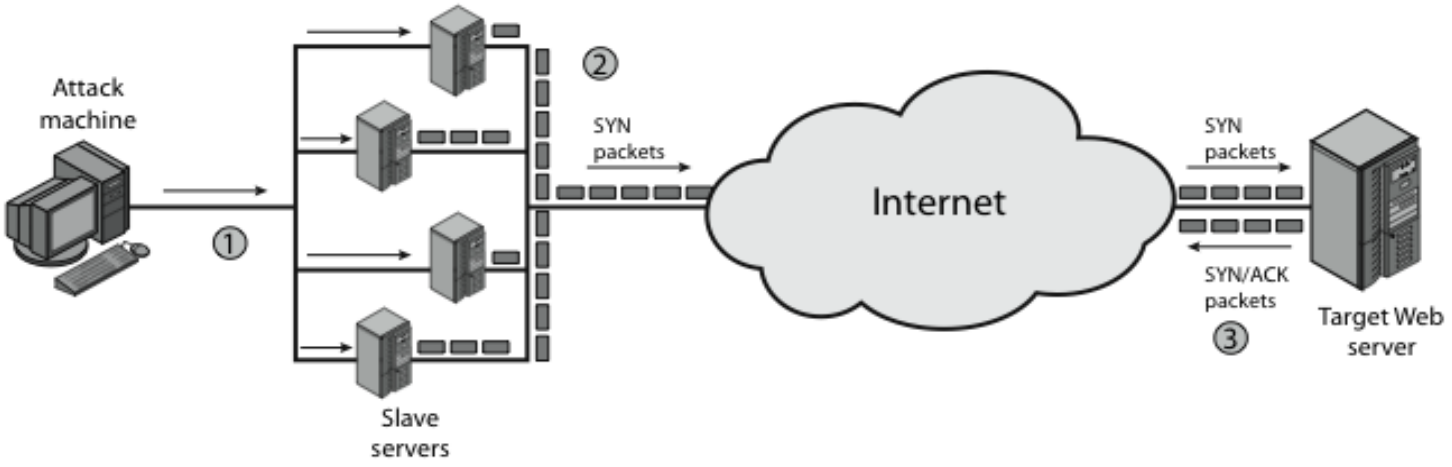
Network-Based Worm Defense



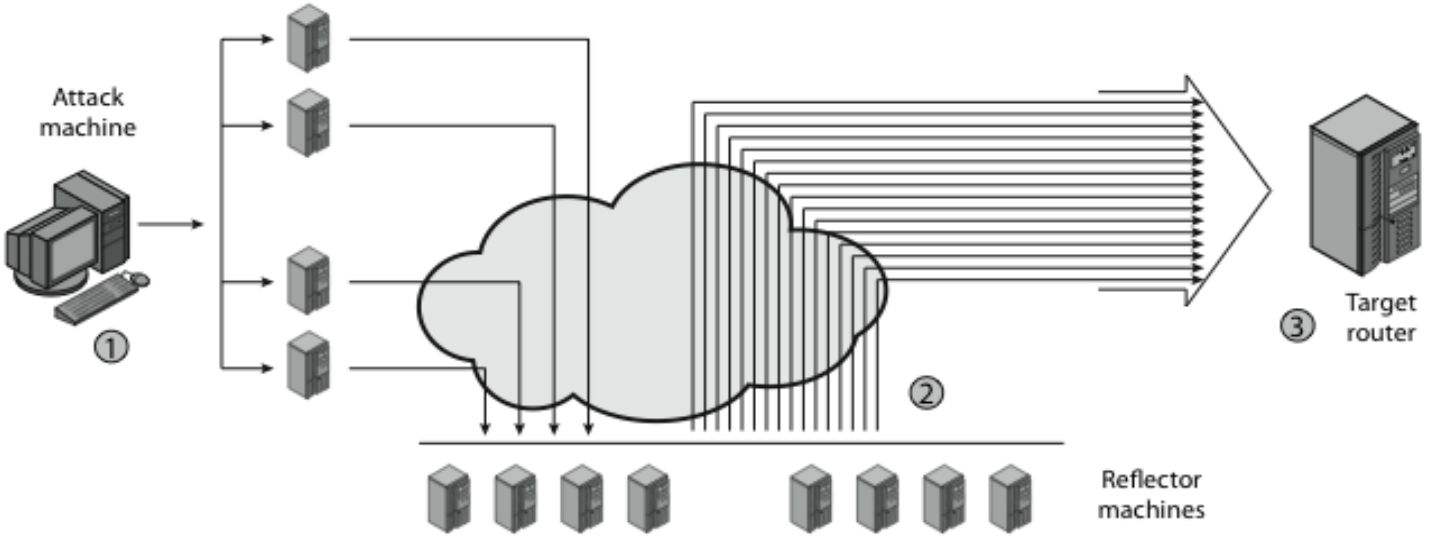
DDoS (Distributed Denial of Services)

- A denial of service (DoS) attack is an ***attempt to prevent*** legitimate users of a service from using that service.
- When this attack comes from a ***single*** host or network node, then it is simply referred to as a ***DoS attack***.
- A more serious threat is posed by a DDoS attack.
- In a DDoS attack, an attacker is able to recruit ***a number of hosts*** throughout the Internet to ***simultaneously*** or in a coordinated fashion launch an attack upon the target.
- This section is concerned with ***DDoS attacks***.

DDoS (Distributed Denial of Services)



(a) Distributed SYN flood attack



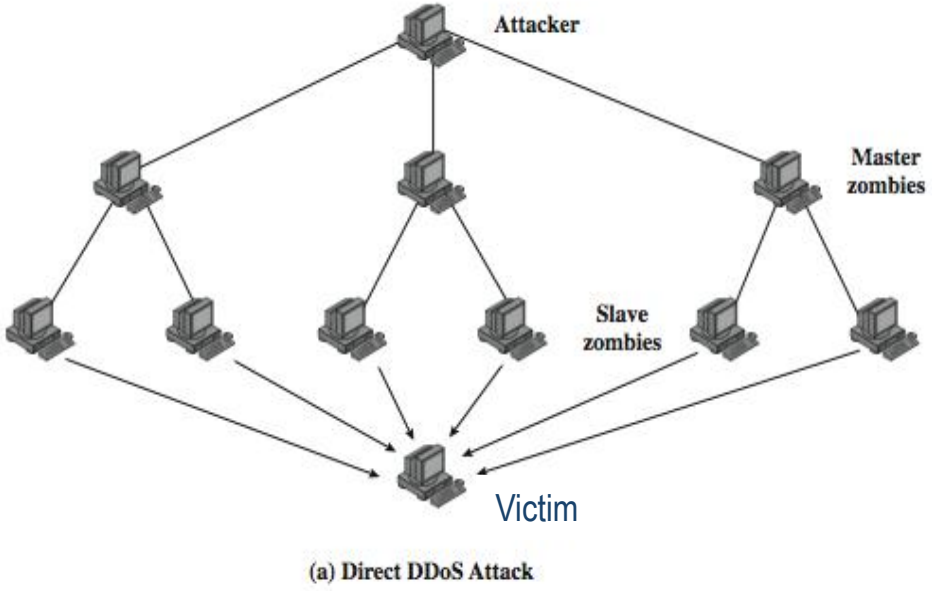
(a) Distributed ICMP attack

DDoS Flood Types

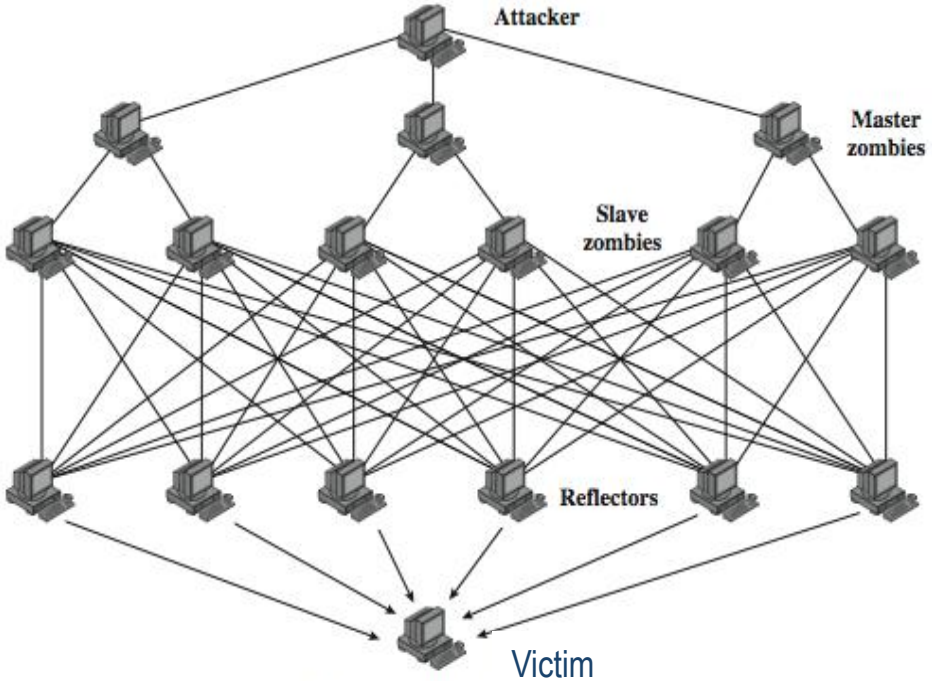
1. The attacker takes control of ***multiple hosts*** over the Internet, instructing them to send ICMP ECHO packets with the target's spoofed IP address to a group of hosts that act as reflectors, as described subsequently.

2. Nodes at the bounce site receive multiple spoofed requests and respond by sending echo reply packets to the target site.

3. The target's router is flooded with packets from the bounce site, leaving no data transmission capacity for legitimate traffic.



(a) Direct DDoS Attack



(b) Reflector DDoS Attack

DDoS Flood Types

- **Direct DDoS attack**
  - the attacker is able to implant zombie software on a number of sites distributed throughout the Internet.
  - Often, the DDoS attack involves ***two levels*** of zombie machines: master zombies and slave zombies.
  - The hosts of both machines have been infected with malicious code

- **A reflector DDoS attack**
  - attack ***adds another layer*** of machines

Constructing the Attack Network

- The first step in a DDoS attack is for the attacker ***to infect a number of machines*** with zombie software
- The essential ingredients in this phase of the attack are the following:
  1. ***Software*** that can carry out the DDoS attack. The software ***must be able to run*** on a large number of machines, ***must be able to conceal*** its existence, ***must be able to communicate*** with the attacker or have some sort of time-triggered mechanism, and ***must be able to launch*** the intended attack toward the target.
  2. ***A vulnerability*** in a large number of systems. The attacker must become aware of a vulnerability that many system administrators and individual users have failed to patch and that enables the attacker to install the zombie software.
  3. ***A strategy*** for locating vulnerable machines, a process known as scanning.

DDoS Countermeasures

In general, there are *three lines* of defense against DDoS attacks:

1. Attack **prevention** and **preemption** (before the attack): These mechanisms enable the victim to endure attack attempts without denying service to legitimate clients. Techniques include enforcing policies for resource consumption and providing backup resources available on demand. In addition, prevention mechanisms modify systems and protocols on the Internet to reduce the possibility of DDoS attacks.
2. Attack **detection** and **filtering** (during the attack): These mechanisms attempt to detect the attack as it begins and respond immediately.This minimizes the impact of the attack on the target. Detection involves looking for suspicious patterns of behavior. Response involves filtering out packets likely to be part of the attack.
3. Attack **source traceback** and **identification** (during and after the attack): This is an attempt to identify the source of the attack as a first step in preventing future attacks. However, this method typically does not yield results fast enough, if at all, to mitigate an ongoing attack.

Summary

- Types of Malicious Software
- Viruses
- Virus Countermeasures
- Worms
- Distributed Denial of Service Attacks

References

1. Cryptography and Network Security, Principles and Practice, William Stallings, Prentice Hall, Sixth Edition, 2013