MONASH University
Information Technology

FIT3031 INFORMATION & NETWORK SECURITY

FIT3031 INFORMATION & NETWORK SECURITY

Lecture 10:

Malicious Software

MONASH University
Information Technology

Unit Objectives

- ✓ OSI security architecture
  - **common security standards and protocols for network security applications**
  - **common information risks and requirements**
- ✓ operation of private key encryption techniques
- ✓ operation of public encryption techniques
- ✓ concepts and techniques for digital signatures, authentication and non-repudiation
- ✓ security threats of web servers, and their possible countermeasures
- ✓ Wireless Network Security Issues
- ✓ security threats of email systems and their possible countermeasures
- ✓ IP security
- ✓ intrusion detection techniques for security purpose
- ✓ risk of malicious software, virus and worm threats, and countermeasures
- firewall deployment and configuration to enhance protection of information assets
- network management protocol for security purpose

Review of Last Lecture

Key points from the last lecture:

- Intrusion on computer infrastructures of big organizations is becoming an increasingly serious problem world-wide
- Early detection of intrusion and deployment of preventive measures are crucial for maintaining the security of the system
- Common intrusion techniques include password cracking, trojan horses, trap doors, etc.
- An Intrusion Detection Systems (IDS) consists of
  - Audit records
  - Detection
  - Response
- Two main approaches for analyzing audit records to detect hacking are
  - Statistical anomaly detection
  - Rule based detection
- CERT has provided a number of guidelines to detect & respond to intrusion
- Good password selection strategy is crucial

Lecture 11: Learning objectives

On completion of this session you should:

- Understand the threat of malicious software
- Be familiar with different types of malicious software
- Describe how trap door, Trojan horses, logic bomb and zombie
- Understand how virus and worm work
- Be familiar with different types of viruses
- Be familiar with Ddos attacks
- Discuss the countermeasures that can be employed to prevent the above attack

Lecture 10: Outline

- **Threats of malicious software**
- **Types of malicious software**
  - trapdoor, logic bomb, trojan horse, zombie
- **Virus attack**
- **Worm attack**
- **Countermeasures**

Malicious Software (1)

- **Perhaps the biggest threat to computer system comes from the malicious intensions to exploit vulnerabilities in O.S and Program software**
- **Numerous incidents of malicious use of software have been reported**
  - many of us are also victim of virus or worm
- **Strong countermeasures are necessary to protect computer systems and information assets**
  - for example, anti-virus software is common
  - needs to install the latest versions

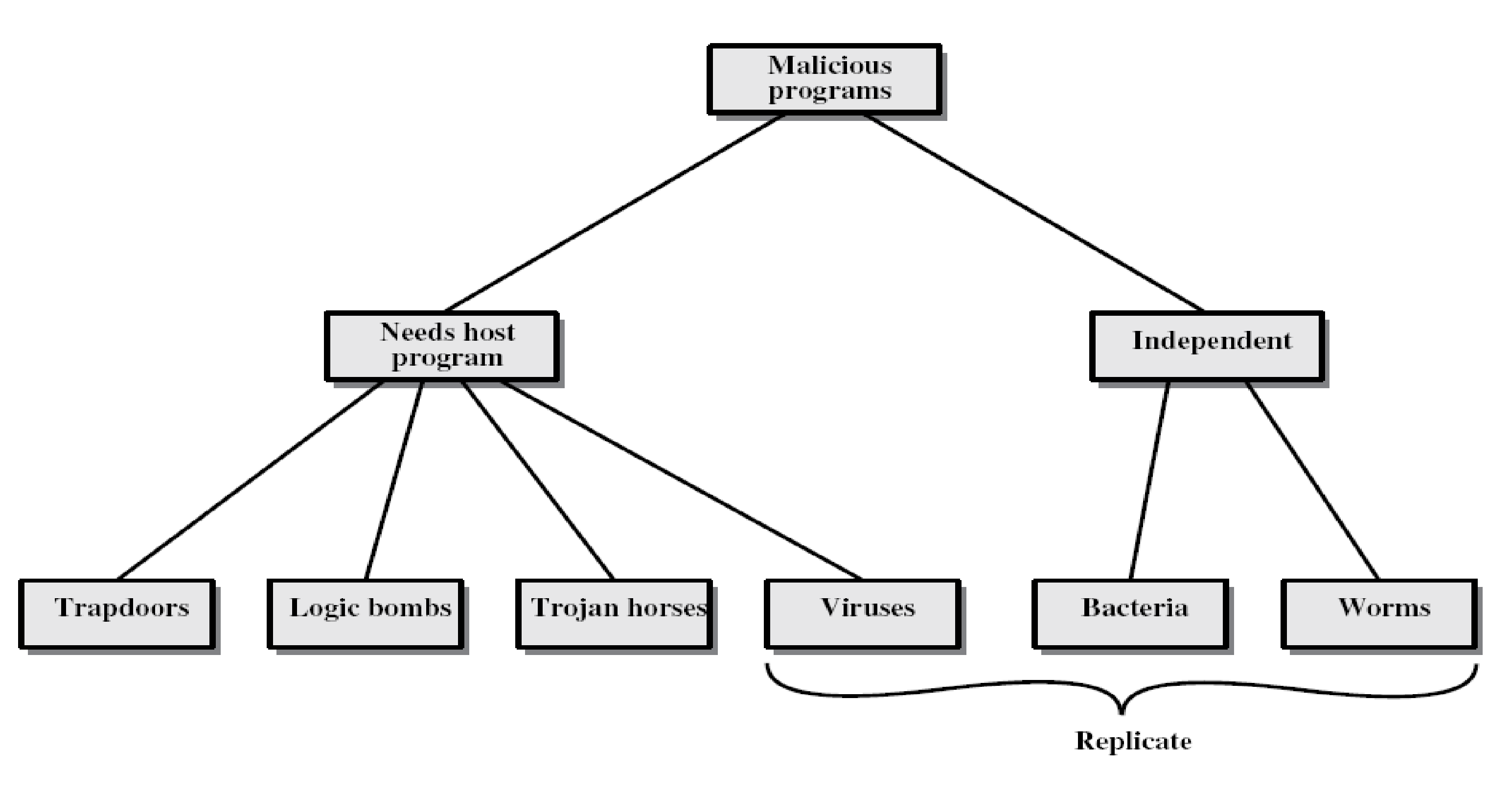
Malicious Software (2)

- **Malicious programs can be broadly categorized into two categories:**
    - **host dependent**
        - cannot exist independently of some application or system program -- parasitic
    - **self contained program**
  - **It can further be categorized as replicating or non-replicating program**
    - **replicating program**
        - when executed, may produce one or more copies of itself to be activated later
    - **non-replicating program**
        - is activated when the host program is invoked to perform a specific function
        - does not make copies of itself

Types of Malicious Software (1)

- Malicious programs can be divided into five types
  - Trap door
  - Logic bomb
  - Trojan horse
  - Virus
  - Worm

Types of Malicious Software (2)



Trap Doors (1)

- Trap doors are left during software development phase
- Usually an application program has an authentication procedure and/or has lengthy setup requiring the user to enter many different values
- For debugging purpose, the developer may leave some code that allows him to gain special privilege to bypass authentication or necessary setup procedure
  - may also want to ensure a method to activate the program if authentication procedure goes wrong

Trap Doors (2)

- A trap door is a secret entry point into a program that allows someone to gain access with going through security access procedures
  - code that recognizes special sequence on inputs, triggered by being run from certain user ID
  - **undocumented** entry points written in **code for debugging**
- It is difficult to implement operating system controls for trap doors
- Security must focus on program development and software update activities

MONASH University
Information Technology

Logic Bomb

- **Malicious code embedded into the program that activates when certain conditions are met**
  - e.g., a particulate date
  - presence/absence of certain file
  - a particular user running the application
- **Logic bomb can alter/delete file, halt a machine or do other types of damage**
- **Example: a logic bomb set by Tim Lloyd cost his employer Omega Engineering huge financial loss**
  - more than 10 million dollar loss
  - layoff of 80 workers

Trojan Horses

- **Trojan horses are useful or apparently useful programs or commands**
  - but contain malicious hidden code
  - do something undocumented which the programmer intended, but that the user would not approve of
  - act as a delivery vehicle
  - which are usually superficially attractive, e.g. game, s/w upgrade, etc.
- **Can cause disastrous consequences**
  - sending data or password to attackers
- **Most anti-virus programs can't detect new Trojans**
  - but once their circulations are reported they can be detected and removed
- **Strategy:**
  - important to know and trust the source of any program before running it

Mobile Code

- **program/script/macro that runs unchanged**
  - on heterogeneous collection of platforms
  - on large homogeneous collection (Windows)
- **transmitted from remote system to local system & then executed on local system**
- **often to inject virus, worm, or Trojan horse**
- **or to perform own exploits**
  - unauthorized data access, root access-compromise
- **Common ways mobile code is spread via**
  - Email attachments, cross-site scripting, interactive/dynamic Web sites
  - Attach to downloads, untrusted software

Multiple-Threat Malware

- **malware may operate in multiple ways**
- **multipartite virus - infects in multiple ways**
  - e.g. multiple file types
- **blended attack uses multiple methods of infection or transmission**
  - to maximize speed of contagion and severity
  - May include multiple types of malware
  - eg. Nimda has worm, virus, mobile code
  - can also use IM & P2P

Virus

- A piece of self-replicating code, i.e., carries code to make copies of itself
  - similar to biological virus
- A virus attaches itself to another program and executes secretly when the host program runs
- The virus loads itself into memory and looks for other programs to infect
- Virus can infect a number of portions of the computer's file system
- A virus is designed specific for an OS and/or hardware platform
  - takes advantage of the weaknesses of particular systems

Virus Life Cycle

A typical virus goes through the following four phases:

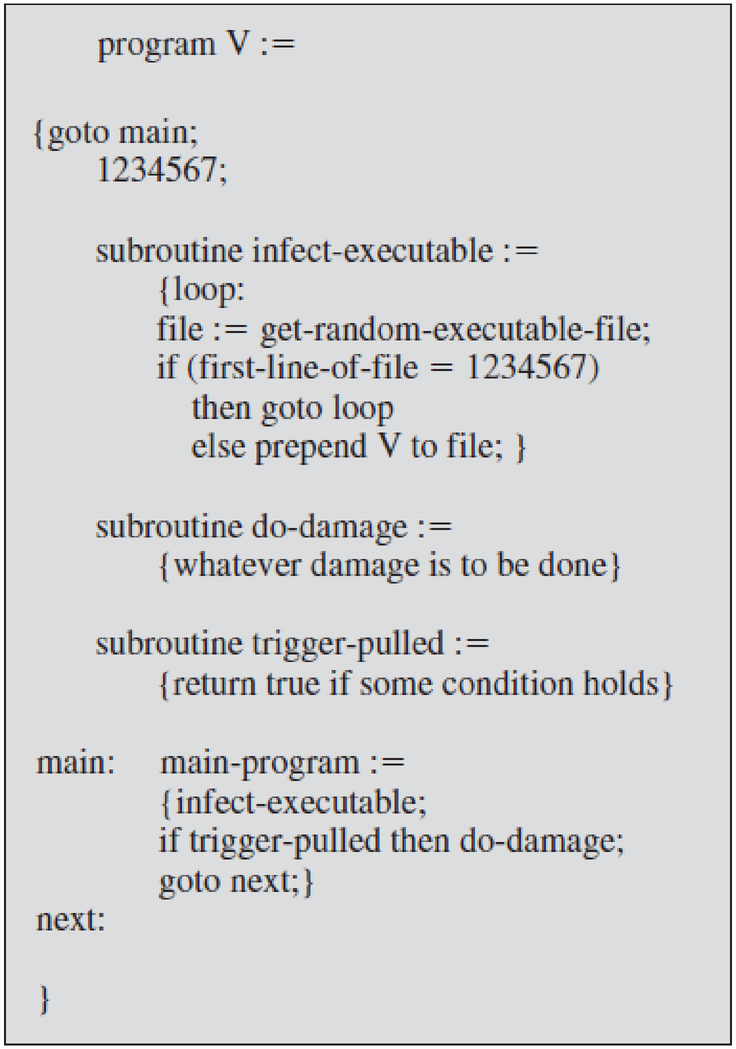
- **Dormant phase:** the virus remains idle waiting to be activated by some events
  - not all viruses have this stage
- **Propagation phase:** the virus embeds a replicated copy into another program or certain system areas on the disk
- **Triggering phase:** the virus is triggered by certain system events
- **Execution phase:** the virus executes causing harmful or harmless action

Virus Structure

- **components:**
  - **infection mechanism** - enables replication
  - **trigger** - event that makes payload activate
  - **payload** - what it does, malicious or benign
- **prepended / appended / embedded**
- **when infected program is invoked, executes virus code then original program code**
- **can block initial infection (difficult)**
- **or propagation (with access controls)**

Virus Operation (1)

- **The first line of the code is a jump to the main virus program**
  - second line is a special marker to check whether the potential victim program has already been infected
- **The infected program is modified so that**
  - when the program is invoked, control is immediately transferred to the main virus program
  - so instead of the proper code running, the virus code runs
- **The virus code becomes active and takes control of the computer**



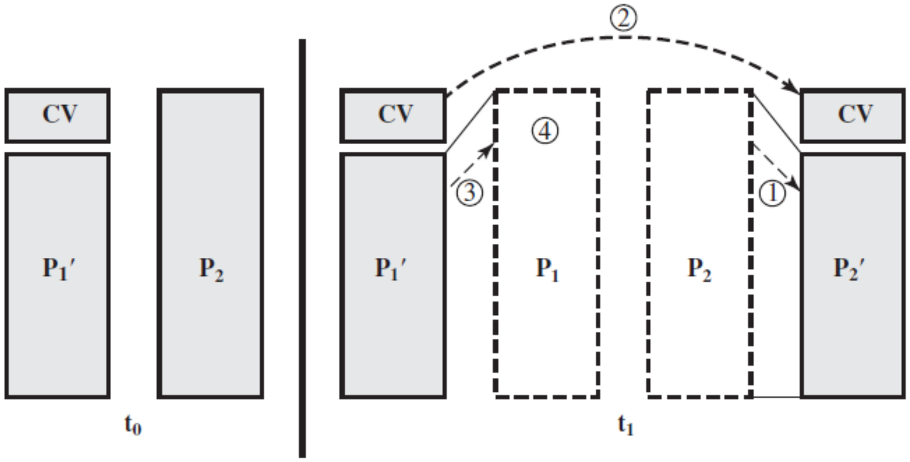
```
     program V :=

{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:

}
```

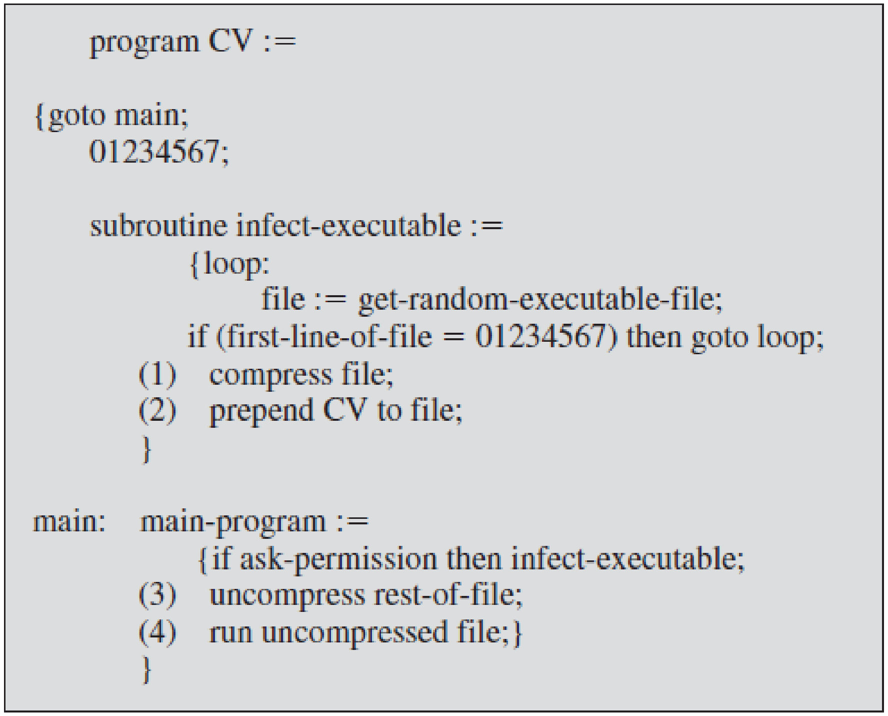
A virus such as this is easily detected because an infected version of a program is longer than the corresponding uninfected one!

Logic for a Compression Virus

A way to thwart such a simple means of detecting a virus is to compress the executable file so that both the infected and uninfected versions are of identical length.

1. For each uninfected file P_2 that is found, the virus first compresses that file to produce P_2’, which is shorter than the original program by the size of the virus.
2. A copy of the virus CV is prepended to the compressed program P_2’.
3. The compressed version of the original infected program, P_1’, is uncompressed.
4. The uncompressed original program is executed.





```
      program CV :=

   {goto main;
      01234567;

      subroutine infect-executable :=
              {loop:
                    file := get-random-executable-file;
              if (first-line-of-file = 01234567) then goto loop;
          (1)    compress file;
          (2)    prepend CV to file;
          }

   main:    main-program :=
               {if ask-permission then infect-executable;
          (3)    uncompress rest-of-file;
          (4)    run uncompressed file;}
          }
```

Virus Operation (2)

There are two ways that a virus behaves when it runs:

- ***direct-action*** viruses
  - executes immediately
  - many file-infector viruses are direct-action
- ***memory-resident*** viruses
  - don't do anything immediately
  - load themselves into memory and wait for a triggering event
  - many file infectors and all boot infectors are memory resident

Virus Classification

- **Virus classification by target**
  - **boot sector**
  - **file infector**
  - **macro virus**
- **Virus classification by concealment strategy**
  - **encrypted virus** – enc with random keys
  - **Stealth virus** – hides against Anti-Virus program
  - **polymorphic virus** – mutates hence signature detection fails
  - **metamorphic virus** – similar to poly, but rewrites & behavior change

Virus Types (3)

- **Boot sector virus:**
  - infects the system area of a disk, e.g., the boot record on disks / hard disks
  - can infect the boot sector of any disk inserted in the machine
  - activates when the user attempts to start up from the infected disk, spread very quickly in an environment where many people share machines
  - because files are no longer commonly transported on floppy-disk, these viruses no longer present a large risk

Virus Types (2)

- **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 Types (3)

Macro virus:

- **became very common in mid-1990s since**
  - platform independent
  - infect documents
  - easily spread
- **exploit macro capability of office apps**
  - executable program embedded in office doc
  - often a form of Basic programming language
- **more recent apps released has protection**
- **recognized by many anti-virus programs**

Virus Types (4)

- **Encrypted virus:**
  - A portion of the virus creates a random encryption key, stored with the virus
  - 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.
  - no constant bit pattern to observe

Virus Types (5)

- **Stealth virus:**
  - explicitly designed to hide from virus scanning programs
  - actively hides any change it has made to the hard disk
  - The virus takes over system functions that are used in reading files or system sectors
- **Polymorphic virus**
  - mutates with every infection, making "signature" detection impossible
  - changes its appearance and size
  - difficult to detect by scanning as each copy looks different
  - needs more than one method of viral detection

Virus Types(6)

- **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.
  - Metamorphic viruses may change their behavior as well as their appearance.

Virus Types(7)

E-Mail virus:

- **more recent development**
- **e.g. Melissa**
  - exploits MS Word macro in attached doc
  - if attachment opened, macro activates
  - sends email to all on users address list
  - and does local damage
- **then saw versions triggered reading email**
- **hence much faster propagation**

Virus Countermeasures

- **prevention - ideal solution but difficult**
- **realistically we need:**
  - detection
  - identification
  - removal
- **if we detect but can’t identify or remove, we must discard and replace infected program**

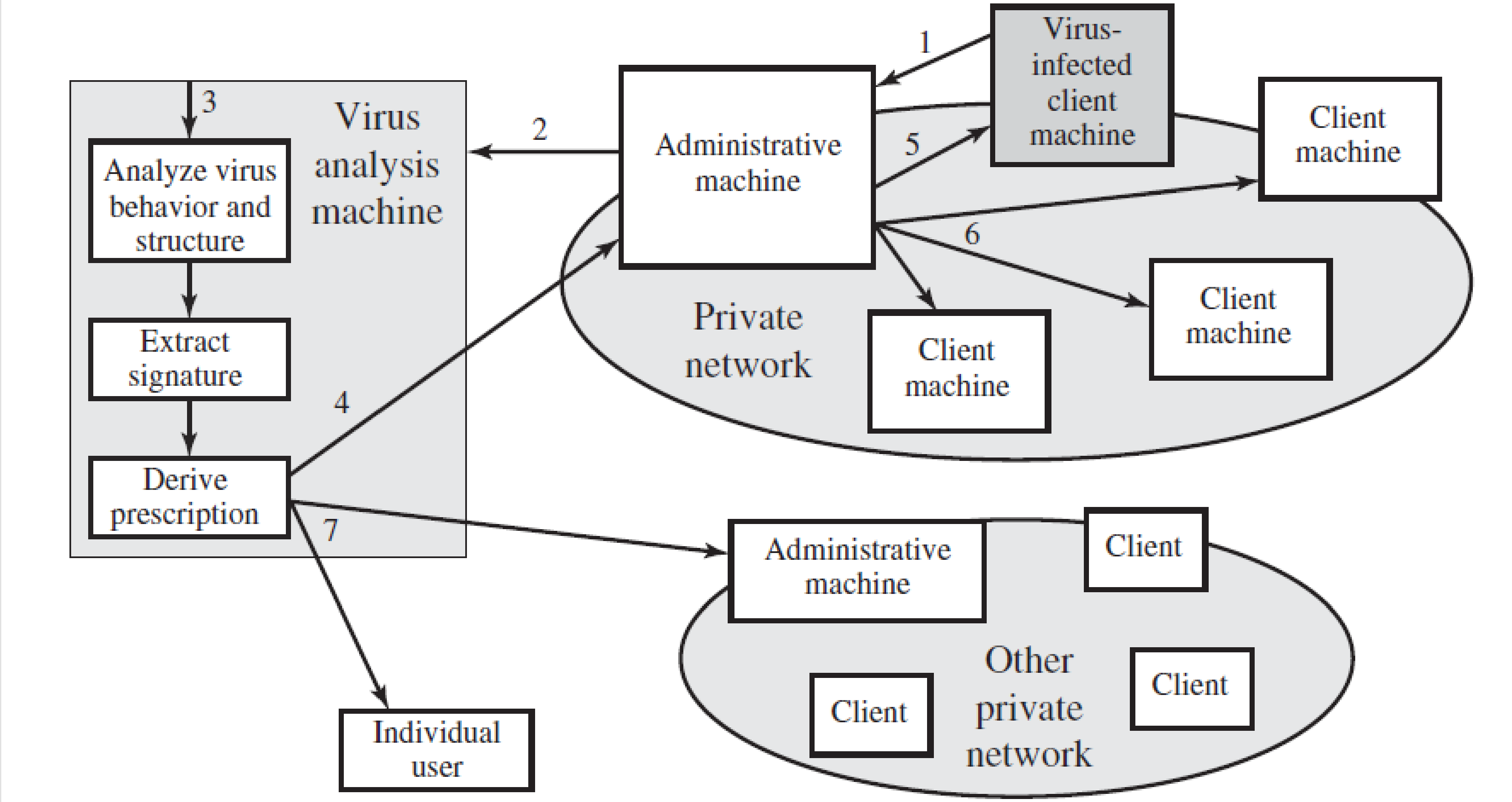
Anti-Virus Evolution

- **virus & antivirus tech have both evolved**
- **early viruses simple code, easily removable**
- **as it become more complex, → so must the countermeasures**
- **generations**
  - first - signature scanners
  - second - heuristics
  - third - identify actions
  - fourth - combination packages

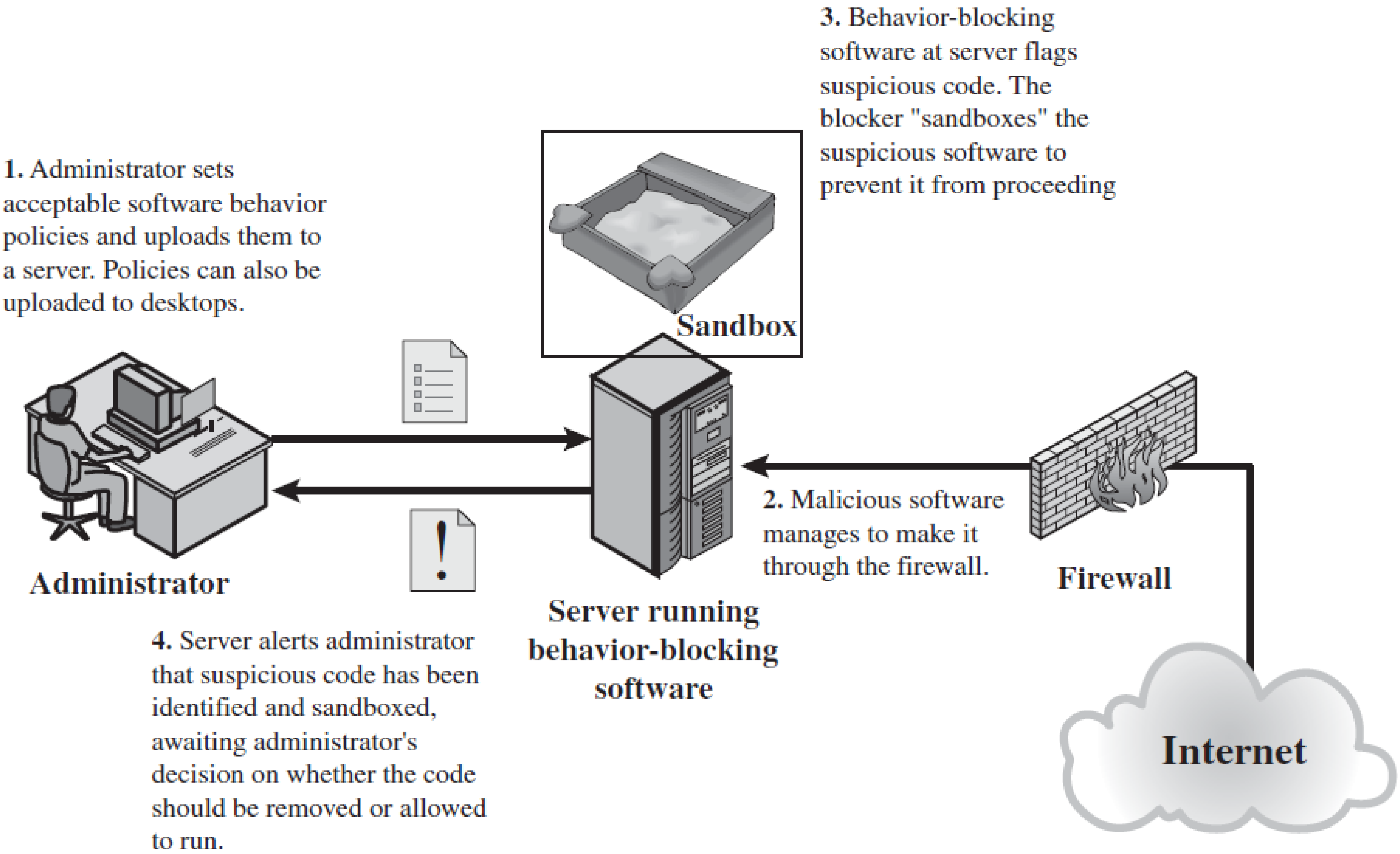
Generic Decryption (GD)

- **Generic decryption (GD) technology**
- **enables the antivirus program to easily detect complex polymorphic viruses & encrypted viruses**
- **runs executable files through GD scanner:**
  - **CPU emulator** to interpret instructions
  - **Virus signature scanner** to check known virus signatures
  - **Emulation control module** to manage process
- **lets virus decrypt itself in interpreter**
- **periodically scan for virus signatures**
- **issue is how long to interpret and scan**
  - tradeoff chance of detection vs time delay

Digital Immune System



Behavior-Blocking Software



1. Administrator sets acceptable software behavior policies and uploads them to a server. Policies can also be uploaded to desktops.

2. Malicious software manages to make it through the firewall.

3. Behavior-blocking software at server flags suspicious code. The blocker "sandboxes" the suspicious software to prevent it from proceeding

4. Server alerts administrator that suspicious code has been identified and sandboxed, awaiting administrator's decision on whether the code should be removed or allowed to run.

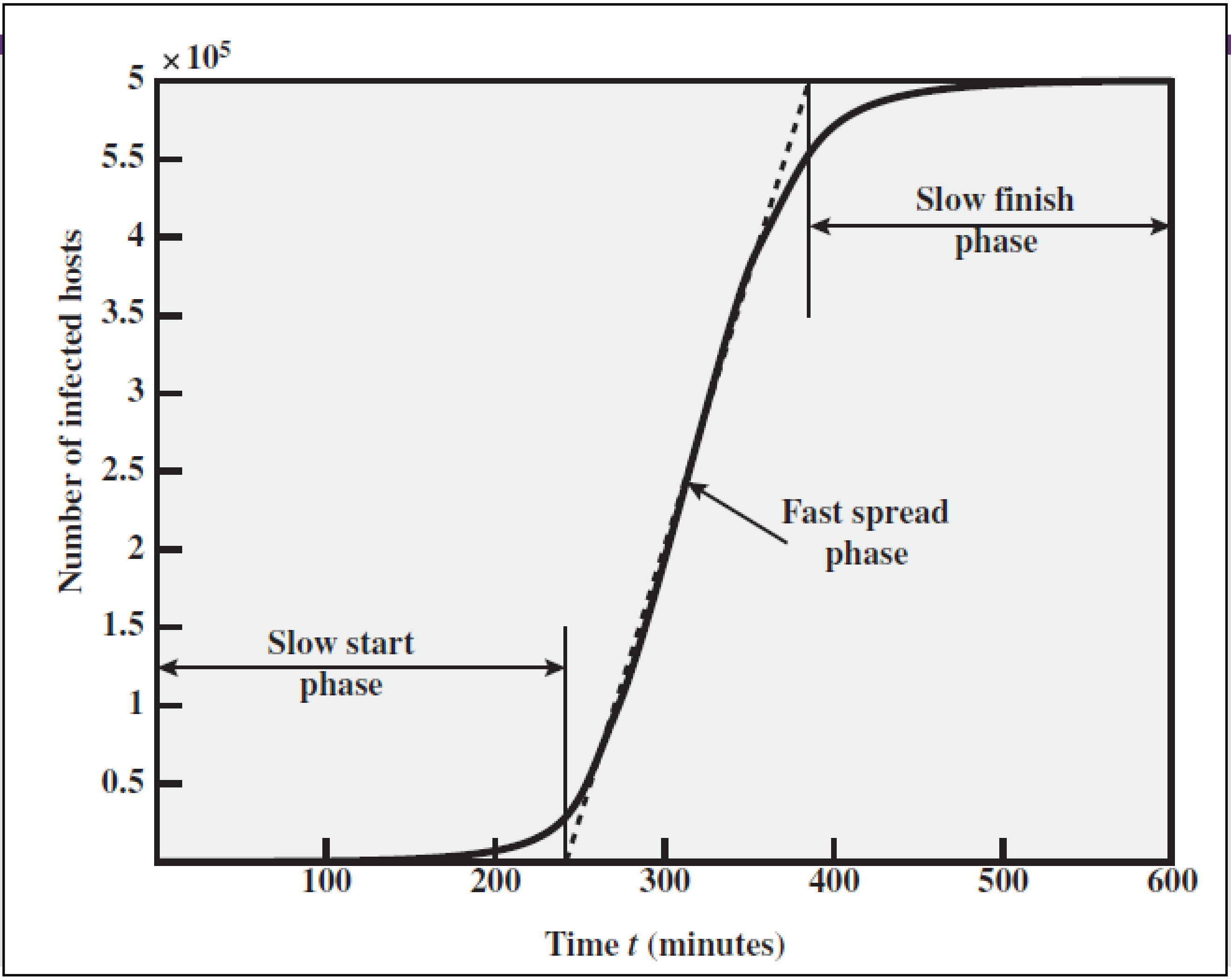
Worms

- **replicating program that propagates over net**
  - using email, remote exec, remote login
- **has phases like a virus:**
  - dormant, propagation, triggering, execution
  - propagation phase: searches for other systems, connects to it, copies self to it and runs
- **may disguise itself as a system process**
- **concept seen in Brunner's "Shockwave Rider"**
- **implemented by Xerox Palo Alto labs in 1980's**

Morris Worm

- **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 Internet Info Server (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

- multiplatform
- multi-exploit
- ultrafast spreading
- polymorphic
- metamorphic
- transport vehicles
- zero-day exploit

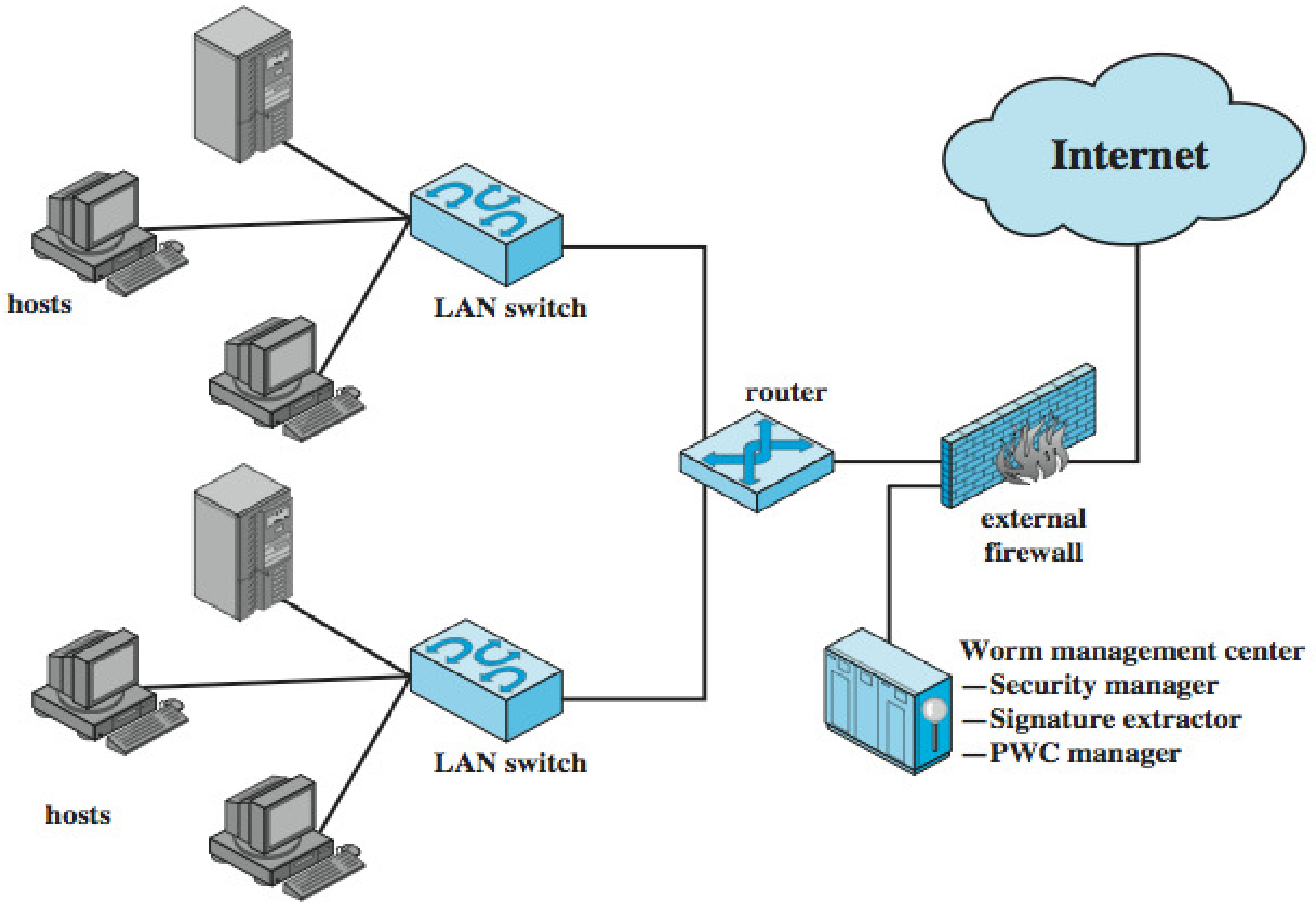
Mobile Phone Worms

- **first appeared on mobile phones in 2004**
  - target smartphone which can install s/w
- **they communicate via Bluetooth or MMS**
- **to disable phone, delete data on phone, or send premium-priced messages**
- **CommWarrior, launched in 2005**
  - replicates using Bluetooth to nearby phones
  - and via MMS using address-book numbers

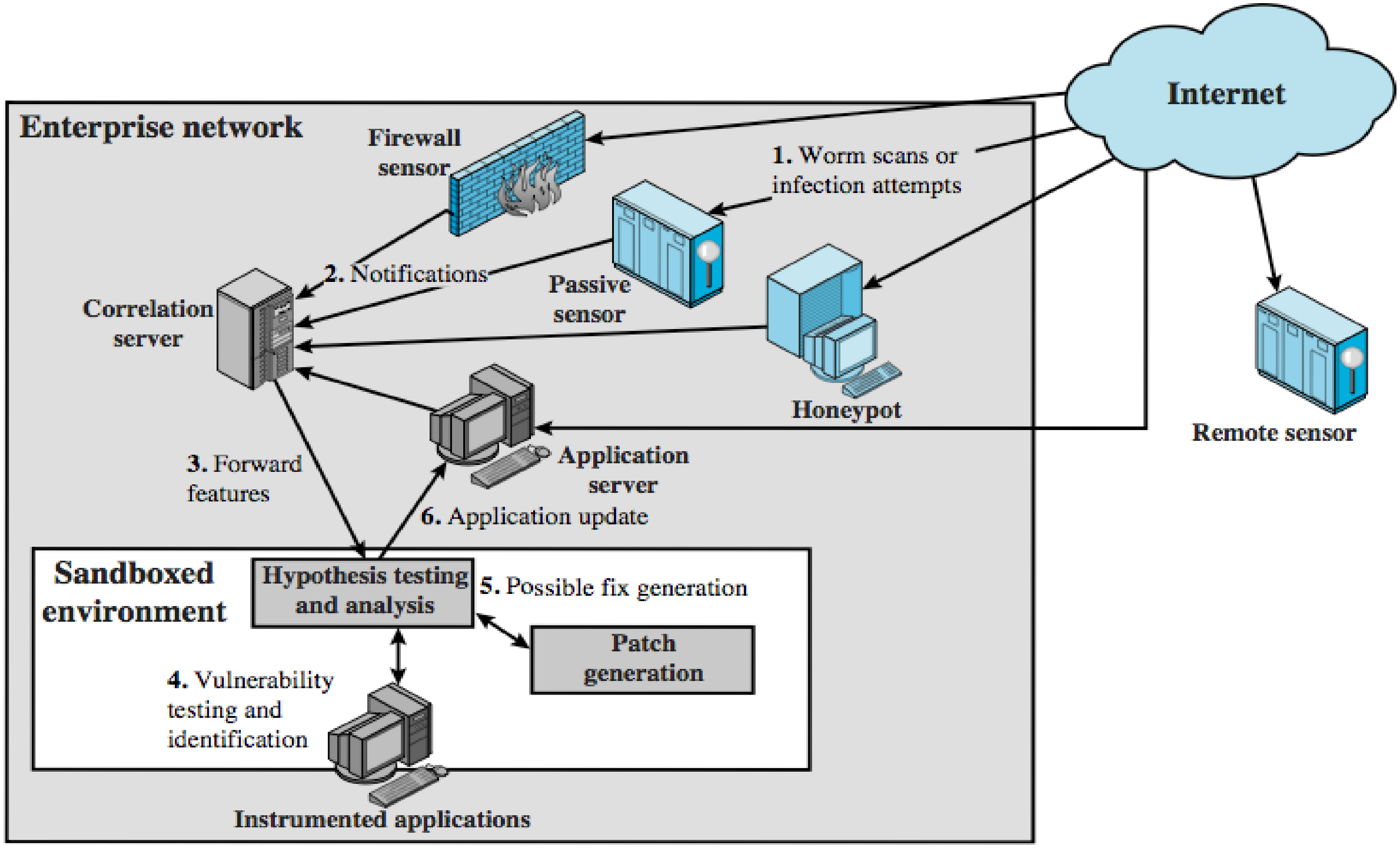
Worm Countermeasures

- **overlaps with anti-virus techniques**
- **once worm on system A/V can detect**
- **worms also cause significant net activity**
- **worm defense approaches include:**
  - signature-based worm scan filtering
  - filter-based worm containment
  - payload-classification-based worm containment
  - threshold random walk scan detection
  - rate limiting and rate halting

Proactive Worm Containment



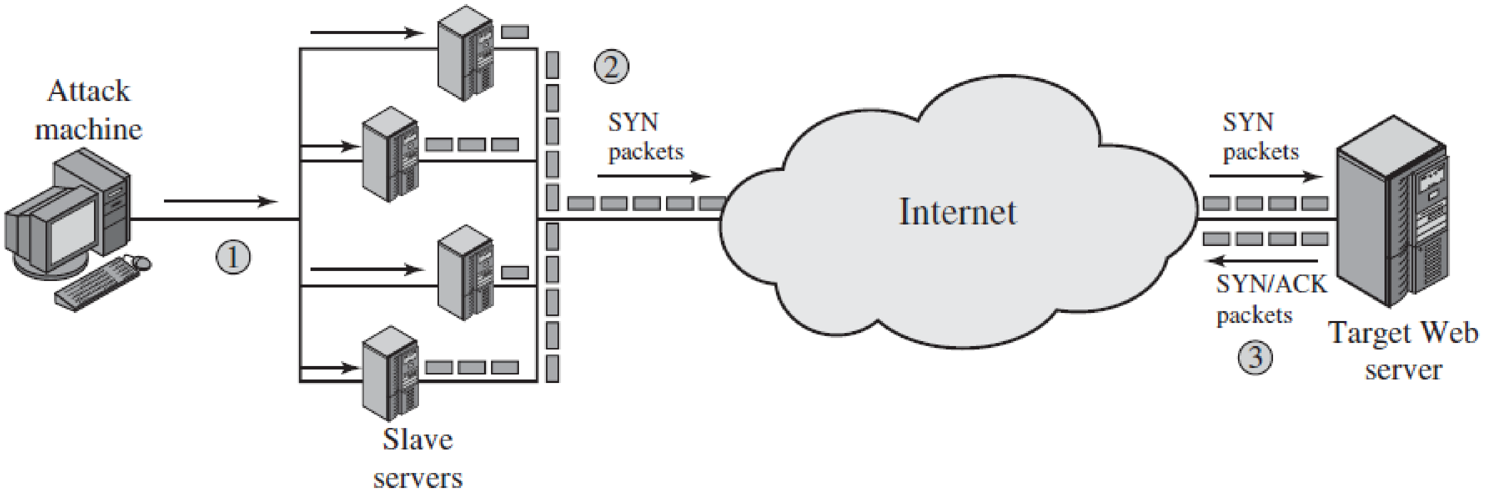
Network Based Worm Defense



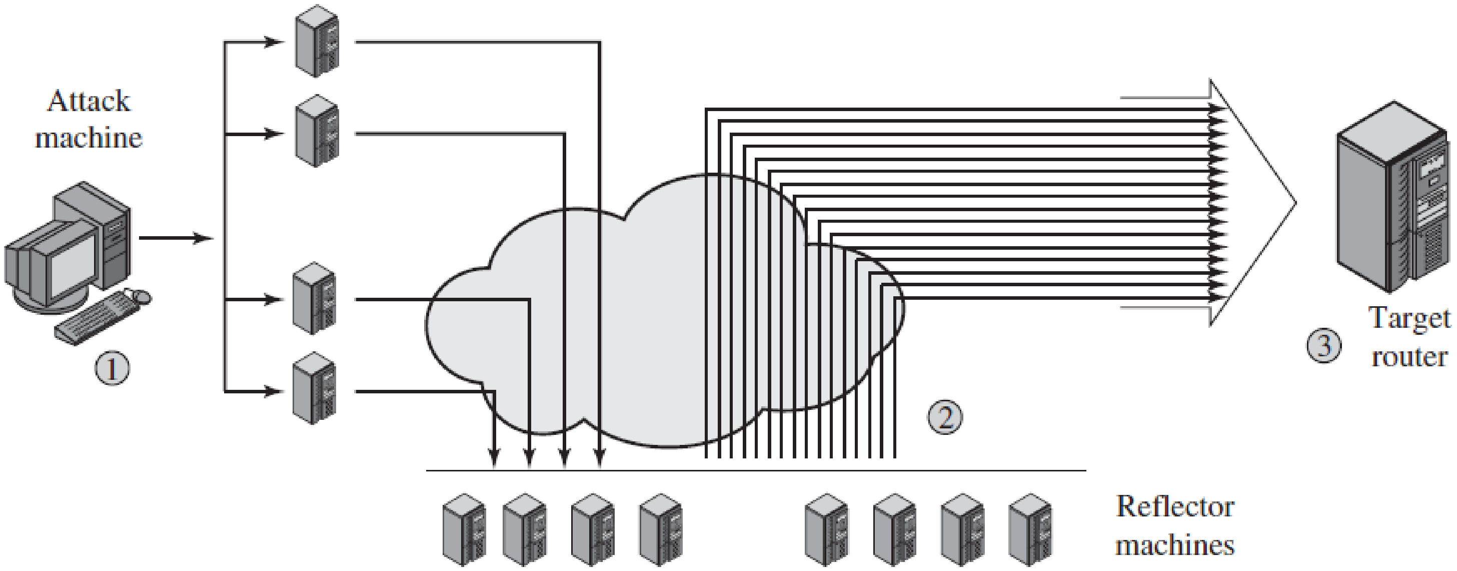
Distributed Denial of Service Attacks (DDoS)

- **Distributed Denial of Service (DDoS) attacks form a significant security threat**
- **making networked systems unavailable**
- **by flooding with useless traffic**
- **using large numbers of “zombies”**
- **growing sophistication of attacks**
- **defense technologies struggling to cope**

Distributed Denial of Service Attacks (DDoS)

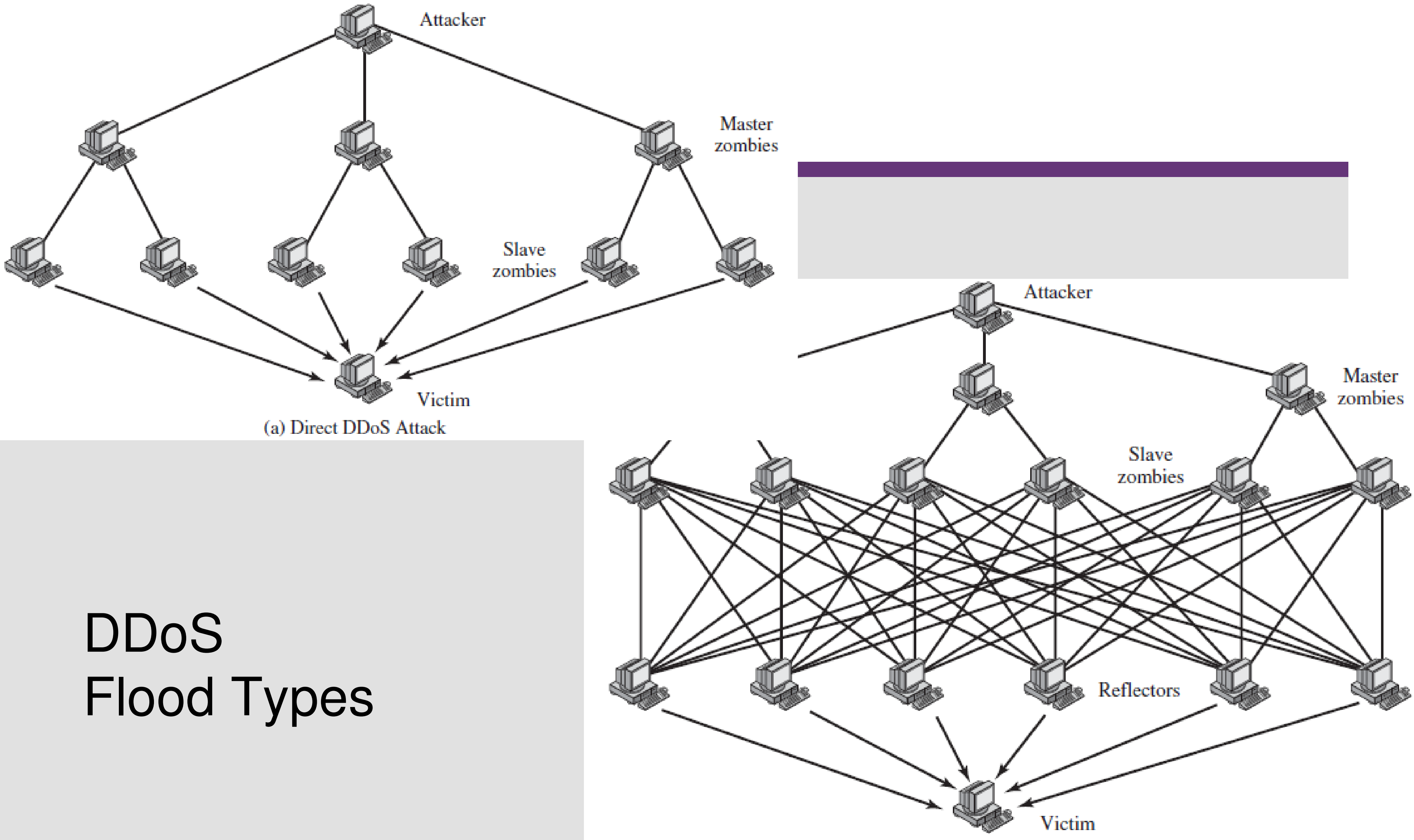


(a) Distributed SYN flood attack



(a) Distributed ICMP attack

DDoS Flood Types



(a) Direct DDoS Attack

(b) Reflector DDoS Attack

Constructing an Attack Network

- **must infect large number of zombies**
- **needs:**
- **software to implement the DDoS attack**
- **an unpatched vulnerability on many systems**
- **scanning strategy to find vulnerable systems**
  - random, hit-list, topological, local subnet

DDoS Countermeasures

- **three broad lines of defense:**
  - attack prevention & preemption (before)
  - attack detection & filtering (during)
  - attack source traceback & ident (after)
- **huge range of attack possibilities**
- **hence evolving countermeasures**

Summary

- **have considered:**
  - – various malicious programs
  - – trapdoor, logic bomb, trojan horse, mobile code
  - – viruses
  - – worms
  - – distributed denial of service attacks

Further Reading

- **Study Guide 10**
- **Chapter 10 of the textbook: Network Security Essentials-Application & Standards" by William Stallings 4th Edition, Prentice Hall, 2011**
- **Additional resources for this week**

- **Acknowledgement: part of the materials presented in the slides was developed with the help of Instructor's Manual and other resources made available by the author of the textbook.**