Intruders and viruses

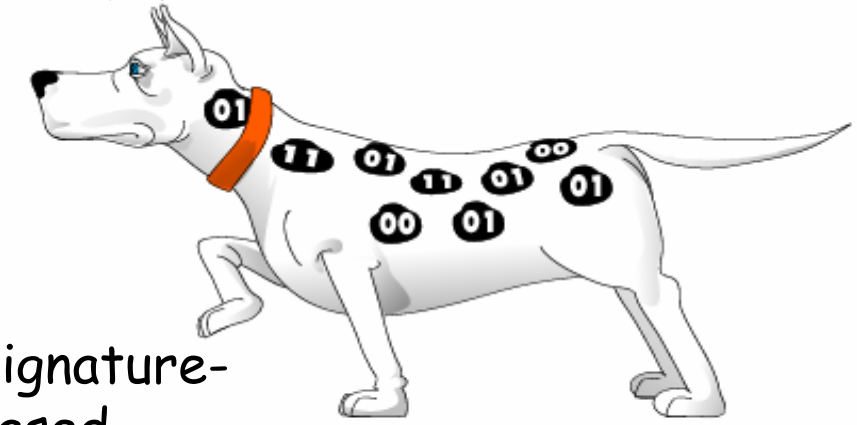
Intrusion Detection Systems

- Firewalls allow traffic only to legitimate hosts and services
- Traffic to the legitimate hosts/services can have attacks
  - CodeReds on IIS
- Solution?
  - Intrusion Detection Systems
  - Monitor data and behavior
  - Report when identify attacks

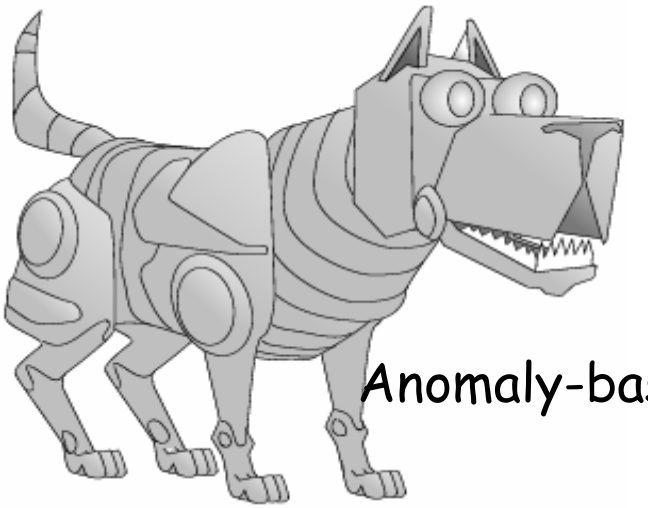
Definition of Intrusion Detection System (IDS)

The art of detecting inappropriate, incorrect, or anomalous activity. ID systems that operate on a host to detect malicious activity on that host are called **host-based ID systems**, and ID systems that operate on network data flows are called **network-based ID systems**.

Types of IDS

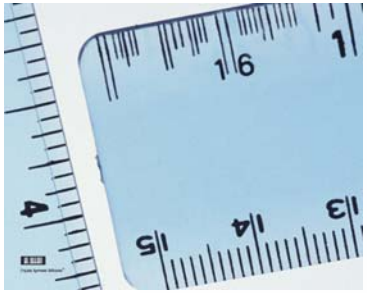


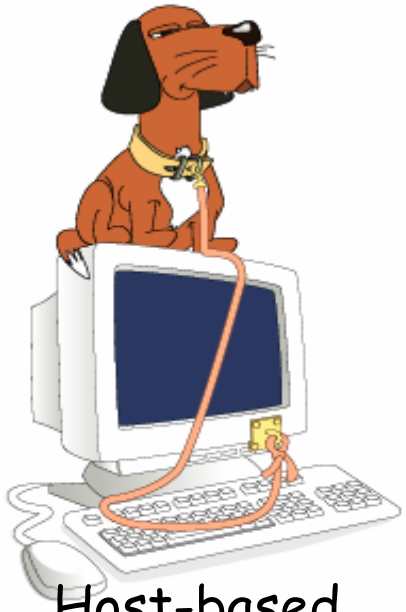
Signature-based



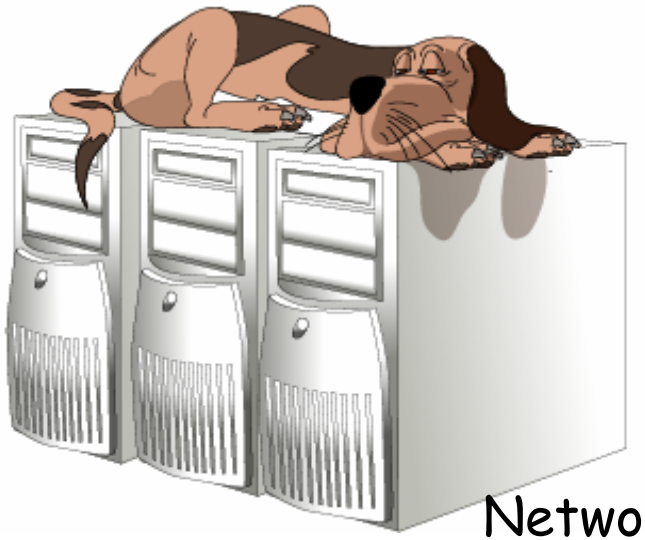
Anomaly-based

Rule - Based





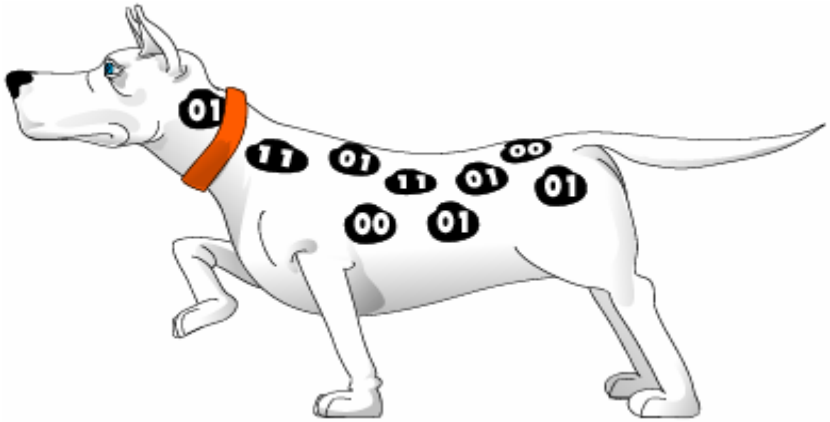
Host-based



Network-based

Signature-based IDS

- **Characteristics**
  - Uses known pattern matching to signify attack



- Advantages?
  - Widely available
  - Fairly fast
  - Easy to implement
  - Easy to update

- Disadvantages?
  - Cannot detect attacks for which it has no signature
  - False positives
  - Maintenance/tweaking
  - Not very hard to evade

Signature-based IDS

- Attack signatures – describe action patterns that may pose a security threat. Typically, they are presented as a time-dependent relationship between series of activities that may be interlaced with neutral ones.

- Selected text strings – signatures to match text strings which look for suspicious action (for example – calling /etc/passwd).

Signature-based IDS

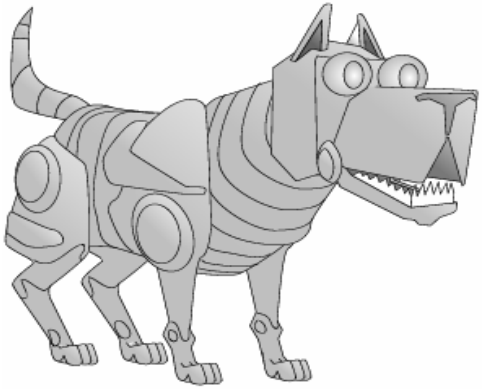
- T A A S 10 20 6668 IRC:XDCC /5Bxdcc/5Dslt
- | | | | | | | | | |
- | | | | | | | | SEARCH STRING
- | | | | | | | EVENT NAME
- | | | | | | PORT
- | | | | | |
- | | | | | COMPARE BYTES
- | | | | |
- | | | | DYNAMIC LOG
- | | | |
- | | | BINARY OR STRING
- | | |
- | | PROTECTED NETWORKS
- | |
- | DIRECTION
- |
- PROTOCOL

Snort has ~1900 signatures
Dragon has ~1700 signatures

http://www.snort.org/docs/

Anomaly-based IDS

- **Characteristics**
  - Uses statistical model or machine learning engine to characterize normal usage behaviors
  - Recognizes departures from normal as potential intrusions
- **Advantages?**
  - Can detect attempts to exploit new and unforeseen vulnerabilities
  - Can recognize authorized usage that falls outside the normal pattern
- **Disadvantages?**
  - Generally slower, more resource intensive compared to signature-based IDS
  - Greater complexity, difficult to configure
  - Higher percentages of false alerts



Anomaly-based IDS

- **Threshold detection**: This approach involves defining the thresholds, independent of users, for the frequency of occurrence of various events. If the count surpasses what is considered a reasonable number that one might expect to occur, then intrusion is assumed.
  - A lot of false positives due to a large difference in behavior of different users.

- **Profile based**: A profile of the activity of each user is developed and used to detect changes in the behavior of individual accounts.

Audit Records used in IDS

Audit records provide input to the profile-based IDS.

Each audit record (Dorothy Denning) contains the following fields:

Subject: Initiators of actions, e.g, users, processes.

Action: operation performed by the subject on or with an object, e.g., login, read, …

Object: receptors of actions, e.g. programs, messages, ..

Exception-Condition: excpetion condition is raised on return

Resource-Usage: amoutn of used resources.

Time-stamp: unigque time-and –date stamp identifying when the action took place.

Metrics Used in Profile-based IDS

- Counter: A count of certain event types is kept over a particular period of time, e.g. number of logins by single user during an hour.

- Gauge: A measure of the current value of some entity, e.g., number of logical connections assigned to a user application.

- Interval timer: The length of time between two related events.

- Resource utilization: Quantity of resources consumed during a specified period, e.g., total time consumed by a program execution.

Tests for IDS based on the metrics

- Mean and standard deviation:
  - Statistical test is to measure the mean and standard deviation of a parameter over some historical period. This gives a reflection of the average behavior and its variability.
- Multivariate:
  - Based on the correlations between two or more variables. Intruder behavior may be characterized with greather confidence.
    - Frequency login and session elapse time
- Markov Process:
  - Establish transtion probabilities among various states
    - Transistion between various commands
- Time Series:
  - Look for events that happens too rapidly or too quickly
- Operational:

Based on a judgement of what is considered abnormal, rather than an automated analysis of past audit records.

Tests for IDS based on the metrics

Measure	Model	Type of Intrusion Detected
Login and Session Activity		
Login frequency by day and time	Mean and standard deviation	Intruders may be likely to log in during off-hours.
Frequency of login at different locations	Mean and standard deviation	Intruders may log in from a location that a particular user rarely or never uses.
Time since last login	Operational	Break-in on a "dead" account.
Elapsed time per session	Mean and standard deviation	Significant deviations might indicate masquerader.
Quantity of output to location	Mean and standard deviation	Excessive amounts of data transmitted to remote locations could signify leakage of sensitive data.
Session resource utilization	Mean and standard deviation	Unusual processor or I/O levels could signal an intruder.
Password failures at login	Operational	Attempted break-in by password guessing.
Failures to login from specified terminals	Operational	Attempted break-in.

Tests for IDS based on the metrics

Command or Program Execution Activity		
Execution frequency	Mean and standard deviation	May detect intruders, who are likely to use different commands, or a successful penetration by a legitimate user, who has gained access to privileged commands.
Program resource utilization	Mean and standard deviation	An abnormal value might suggest injection of a virus or Trojan horse, which performs side-effects that increase I/O or processor utilization.
Execution denials	Operational model	May detect penetration attempt by individual user who seeks higher privileges.

Tests for IDS based on the metrics

File Access Activity		
Read, write, create, delete frequency	Mean and standard deviation	Abnormalities for read and write access for individual users may signify masquerading or browsing.
Records read, written	Mean and standard deviation	Abnormality could signify an attempt to obtain sensitive data by inference and aggregation.
Failure count for read, write, create, delete	Operational	May detect users who persistently attempt to access unauthorized files.
File resource exhaustion counter	Operational	

Rule-based IDS

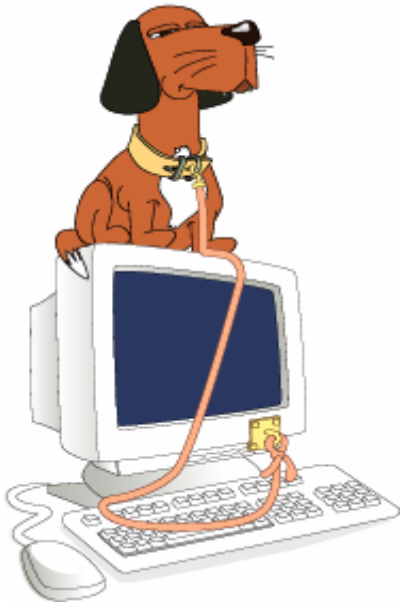
- Historical audit records are analyzed to identify usage patterns and to generate automatically rules that describe the patterns.
- Rules may represent past behavior patterns of users, programs, privileges, …
- Current behavior is then observed, and each transaction is matched against the set of rules to determine if it conforms to any historically observed pattern of behavior.

Network-based IDS

- Characteristics
  - NIDS examine raw packets in the network passively and triggers alerts
- Advantages?
  - Easy deployment
  - Unobtrusive
  - Difficult to evade if done at low level of network operation
- Disadvantages?
  - Fail Open
  - Different hosts process packets differently
  - NIDS needs to create traffic seen at the end host
  - Need to have the complete network topology and complete host behavior

Host-based IDS

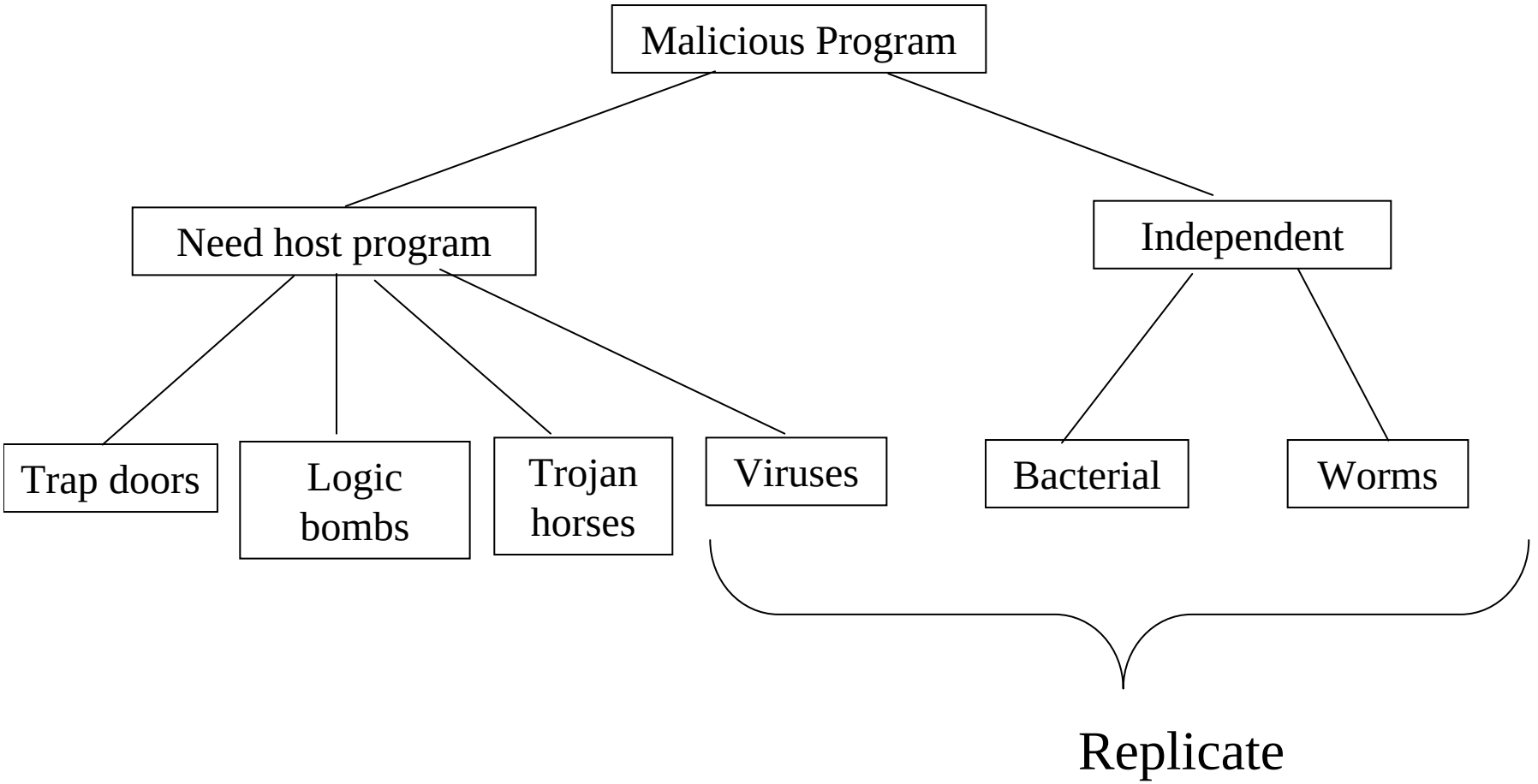
- Characteristics
  - Runs on single host
  - Can analyze audit-trails, logs, integrity of files and directories, etc.
- Advantages
  - More accurate than NIDS
  - Less volume of traffic so less overhead
- Disadvantages
  - Deployment is expensive
  - What happens when host get compromised?



Viruses

- Virus is the common term to describe malicious programs.

Taxonomy of Malicious Programs



Trap Doors

- A secret entry point into a gprogam that allows someone that is aware of the trap door to gain access without going through the usual security access procedures.
- Used legitimately for many years by programmers to debug and test programs.
- Become threats when they are used by unscrupoulus programmers to gain unauthorized access.

Logic Bomb

- Oldest types of program threats

- Coded embedded in some legitimate program that is set to "explode" when certain conditions are met.
  - Particular day of the week
  - Famous cases: employee ID number, library systems

Trojan Horses

- Program or command procedure containing hidden code that when invoked, performas some unwanted or harmful functions.

- Gain access to files of another user on a shared system by changing permission when the unawared user run the Trojan horse program disguised as the normal program.
  - ls, ps

- Data destruction

Viruses

- A virus is a program 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.

- We will discuss shortly in details.

Worms

- Network worm programs use network connections to spread from system to system.
  - Electronic mail: A worm mails a copy of itself to other systems.
  - Remote execution capability (rcp): A worm executes a copy of itself on another system.
  - Remote login capability:A worm logs onto a remote system as a user and then uses command to copy itself from one system to the other.
- Worm can behave as a computer virus or bacteria or it could implant Trojan horse programs or perform any number of distruptive or destructive actions.

Bacteria

- Bacteria are programs that do not explicitly damage any file.
- Typical bacterial program dupicate itself simultaneously, or create new files, each of which is a copy of the original source file of the bacterial program.
- The process continues and eventually takes up all the processor capacity, memory, or disk space, denying users access to those resources.

Nature of Viruses

- Typical virus goes through the following four stages:
  - Dormant phase: Virus is idle.
    - Activated by some event, such as a date.
  - Propagation phase: places an identical copy of itself onto other programs or into certain system areas on the disk.
  - Triggering phase: The virus is activated to perform the function for which it was intended.
    - Activated by a varity of system events.
  - 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 program and data files.

Structure of a simple virus

- A virus can be prepended or postpended to an executable program.
- When the infected program is invoked, it will first execute the virus code and then execute the original code of the program.

Structure of a simple virus

```
        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:

}
```

Detecting simple virus

- It is easy to detect the simple virus by simply comparing the size of the original and the infected program.

Compression virus

1. For each uninfected file P2 that is found, the virus first compresses that file to produces P2', which is shorter than the original program by the size of the virus.

2. A copy of the virus is prepended to the compressed program.

3. The compressed version of the original infected program, P1', is uncompressed.

4. The uncompressed original program is executed.

Structure of a compression virus

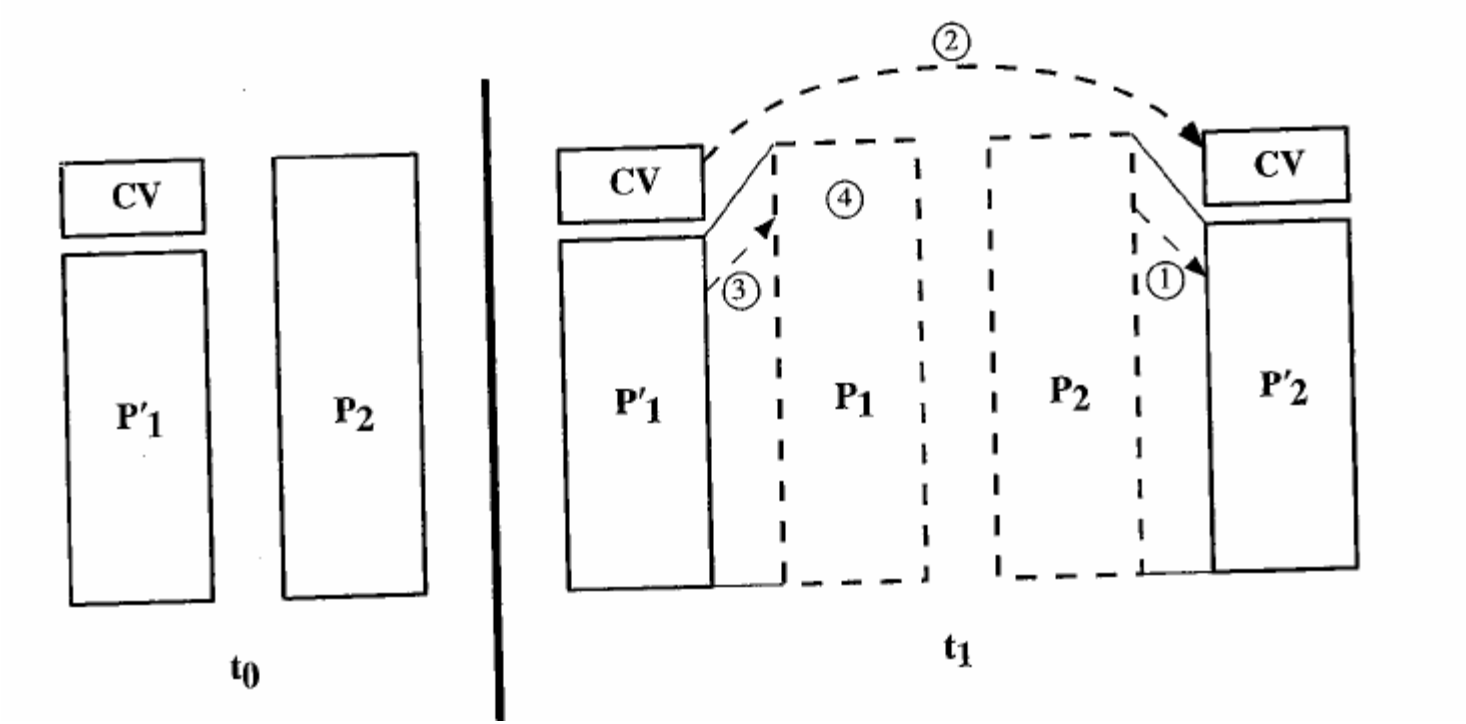
```
        program CV :=

{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;}
                        }
```

Structure of a compression virus



Types of Viruses

- Parasitic virus: Most common form of virus. A parasitic virus attaches itself to executable files and replicates, when the infected program is executed, by finding other executable files to infect.
- Memory-resident virus: Lodges in main memory as part of a resident system program. From that point on, the virus infects every program that executes.
- Boot sector virus: Infects a master boot record or boot record and spreads when a system I booted from the disk containing the virus.
- Stealth virus: A form of virus explicitly designed to hide itself from detection by antivirus software.
- Polymorphic virus: A virus that muates with every infection, making detection by the "signature" of the virus impossible.

Macro-viruses

- Platform independent, hence spread quickly.
- Macro virus infect documents, not executable portions of code.
- Very easy to spread, usually by electronic mail.

Macro-viruses

- In Microsoft word:
  - Autoexecute: if a macro named AutoExec is in the "normal.dot" template or in a global template stored in Word's start up directory, it is executed whenever Word is started.
  - Automacro: An automacro executes when a defined event occurs, such as opening or closing a document.
  - Command macro: If a macro in a global macrofile or a macro attached to a document has the name of an existing Word command, it is executed whenever the user invokes that command (e.g File Save).

Antivirus Approaches:

- Detection
- Identification
- Removal

- First-generation: simple scanner:
  - Identify signature of a virus

- Second-generation: Heuristic rules to search for probable virus infection.
  - Looks for fragments of code that are often associated with virus.
  - E.g. encryption loop in compression virus.

- Third-generation: Program are memory-resident which actively identifiy a virus by its actions rather than its structure in an infected program.

- Fourth-generation: contain a mix of first, second, and third generations.