Viruses are programs that insert themselves into one or more files. Usually they will then perform some action (but they don't have to - some viruses will just propagate, presumably just to prove that they could)

Viruses have two phases: the insertion phase (where the virus infects the file) and the execution phase (where the virus executed its payload, if it has one).

When we talk about viruses we typically talk about three different types: file infectors, boot-sector viruses, and multi-partite viruses.

File Infectors

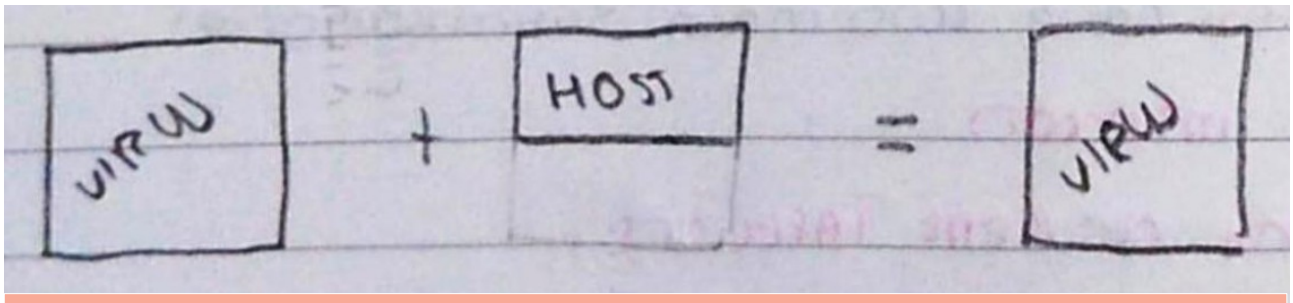
File Infector Viruses are the viruses that attach themselves to executable files (which also includes files like word documents which can contain executable code in the form of macros).

File infector viruses have two main subcategories: Direct Infectors and Memory Resident Infectors.

Direct infectors go straight for the file – infecting it directly They can do that in a number of ways.

Direct infector – overwriting viruses overwrite sections of the original program with the virus code.

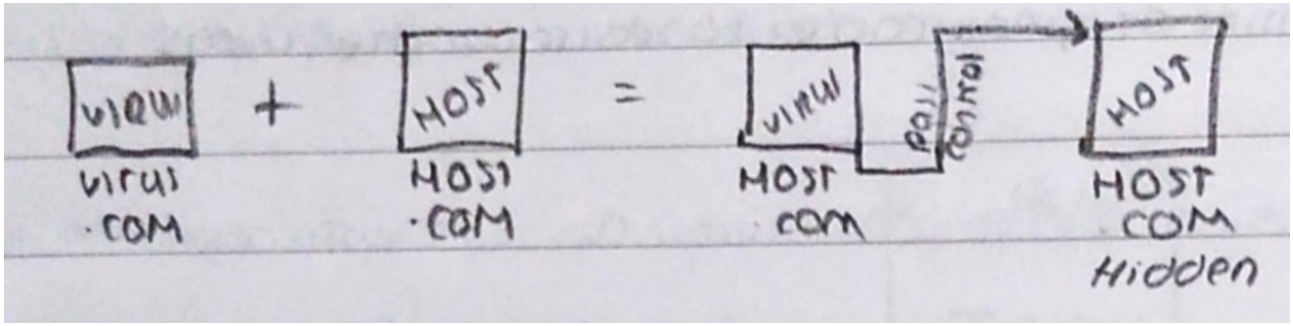
These viruses may break the original program (since you're blindly overwriting lines of code). If the original file (the host file) is smaller than the virus file, then the host file will be completely overwritten and the resulting file will be the same size (and an exact copy) of the virus file. Obviously this would be easy to detect, so authors tend to keep their virus files as small as possible.

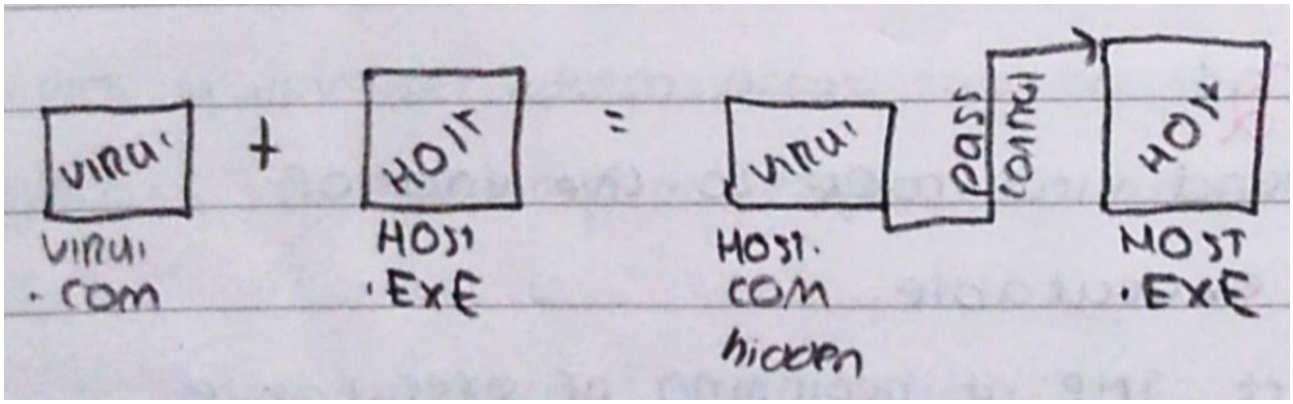


Direct infector – companion viruses aim to be called first and then pass control over to the infected file so that the user of the system is none the wiser

From the users perspective, they've executed a file and that file *does* run. It just so happens that the virus ran first.

It'll do this by taking advantage of execution orders. In DOS (the windows operating system) if the extension of a file to run is not given, then the OS executes in a predetermined order (COM files, then EXE, then BAT). This means that if the file the host intends to run is called 'executeme.exe' and the virus names itself 'executeme.com', when the user tries to run 'executeme' it's the virus version that will run first. The virus might also make use of hidden files, so if the original host file is called 'executeme.com' the virus can set it to hidden, make it's own 'executeme.com', and so the virus will run first (non hidden files take precedent).

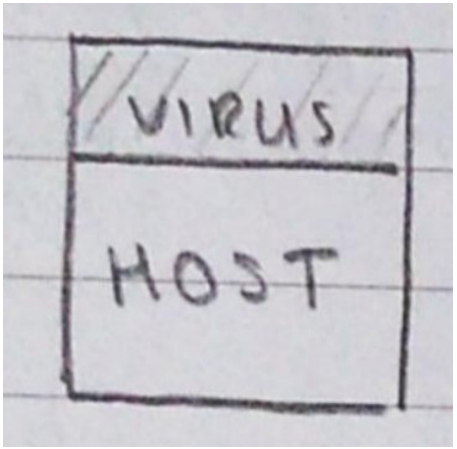




Direct infector – parasitical viruses edit the code of the host file so that the virus is hidden amongst the original code (without breaking the original code).

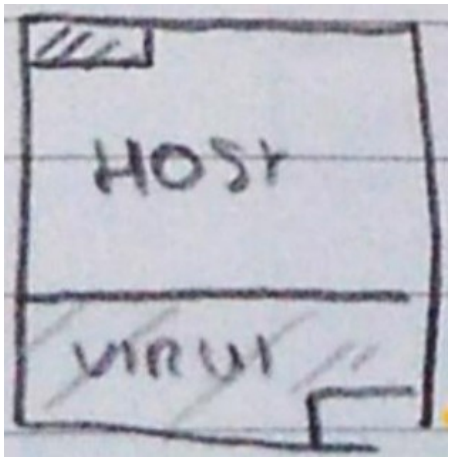
There is the **pre-pending** approach:

This is where the virus code is put at the beginning of the executable. The original code is just shifted.



There is the **appending** approach:

This where we attach the virus code to the very end of the host file. We also need to put a jump (JMP) instruction at the very beginning of the host file so it executes the virus first. The virus code also then needs to have a jump instruction back into the host file.



There is the **fragmenting** approach:

This is where the virus code is intermingled throughout the original code and we use a lot of jump instructions to execute all the different parts.

Memory resident viruses hide and wait in memory until a host program is executed and then infects it.

These can be harder to detect since you can't analyse a static file to find the virus.

Boot Sector Viruses are viruses that target the boot sector so that they can execute when the system is first turned on. Usually they will target the Master Boot Record or Partition Boot Sector instructions that are executed during the boot-up sequence.

The boot sector is the part of a disk that is used to initially launch a system. It's the first thing that runs when a computer is turned on. A boot sector virus will typically hijack control but inserting a jump instruction at the very beginning of the boot sector's code and, once the virus has finished executing, passes control back to the boot sector code so that the system appears to launch normally.

Boot sectors don't tend to have a lot of space to hide virus code (usually they are only 512 bytes of code) so a boot sector virus will normally write parts of its code to other sectors of the disk as well.

Multi-Partite Viruses are viruses that will infect both files and the boot-sector.

Infecting File Types

Viruses can target many file types for different purposes.

Most common is probably executable programs (so the virus will execute when the infected program is run, and usually the virus runs first).

Device drivers are also a popular target (device drivers run in something called ‘kernel mode’ - which essentially means that viruses running here will have elevated privileges which will allow them more access rights on the system).

Archives can also be a target (these will be trojan horses in ZIP files, where the user is socially engineered to unzip the virus).

Dynamically Linked Libraries are especially useful targets for viruses since these libraries are called by many other programs (so there is a high chance of the virus code being executed).

Macro viruses are viruses that are attached to file types like word documents. How effective they are depends on the expressibility of the macro language and it’s resource access. Microsoft office for instance has a macro language called VBA (visual-basic, application specific). These types of viruses can be particularly harmful in corporate environments seeing as word documents are frequently passed around in emails and opened without thought (since a lot of people aren’t aware of the potential for macro viruses). This is a ripe target for social engineers.

Script viruses are similar to macro viruses in that they are defined by the language they are written in (interpreted rather than compiled). In fact many viruses, like those written in VBA, are both macro and script viruses. So the distinction isn’t all that clear.