Harmless and Useful Viruses Can Hardly Exist

Pavel Lamačka, Bratislava, Slovak Republic, www.hisys.sk
(Proceedings of The Fifth International Virus Bulletin Conference, Boston September 1995, p.193-198.)

Introduction

Some virus authors and even some antiviral experts [Cohen-85] [Cohen-92] claim that not all viruses and programming techniques used in them are harmful and therefore bad. They argue that viruses which have the ability to execute no action, neither harmful nor useful, are harmless and therefore neutral, and that viruses which are able to execute beneficial actions are useful and therefore good. Discussions about harmless and useful viruses are still not finished as can be seen, for example, from [Kaspersky-93] or [Timson-93]. Neither they are academic, because our basic attitude towards viruses, the techniques of their implementation, and their originators and propagators, depends on the results of these discussions. If viruses are really neutral in nature, it is necessary to discipline only those responsible for their unsuitable purposes and usage. But if we find out that viruses are bad in principle, we obtain the right to take a consequently defensive attitude towards their originators and propagators.

The goal of this paper is the presentation of reasoning leading to a standpoint which is usable in practice regarding the existence and feasibility of harmless and useful computer viruses. The presented reasoning is based on a combination of known, lesser known and so far probably undiscussed facts and conclusions. Those of which are considered contributions of this paper, ***are indicated***.

Harmful viruses

Before we start a discussion about the possible existence of harmless and useful viruses, we will take a look at harmful viruses. Viruses which are able to execute a harmful action, like destroying data or disabling the usage of a computer, are considered harmful. Generally a harmful virus V_H (Fig.1) consists of at least the two following modules: a module of self-replication M_{SR} and a module of harmful action M_{HA}. The harmful action is usually executed by the virus on a certain condition. Other modules and functions of the viruses, for example, stealth, encryption or polymorphism, will not be considered here, because they are irrelevant to this paper.

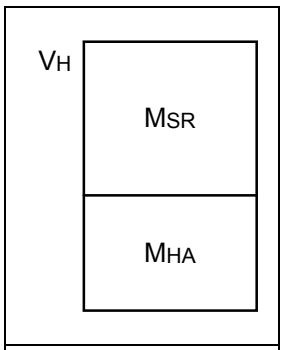

Fig.1 Harmful virus

Many people claim that viruses have gained their bad reputation only due to the harmful actions which many of them execute. Let us therefore look at whether the harmfulness of a virus would disappear after removing the harmful action code from it, and then at whether it is possible to obtain a useful virus after it is given the ability to execute a useful action.

Harmless viruses

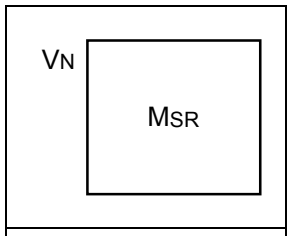

Fig.2 No action virus

Virus VN (Fig.2), which can do nothing but self-replicate, consists of a self-replication module MSR only. Several such viruses are known in practice. It is known that although this type of virus does not contain any harmful action module, it is able to damage the code of a program on which it is a parasite. This is often caused by the untidy implementation of the virus. It may happen that the virus is implemented in a competent way, but then it meets with a new program structure which it cannot infect properly and therefore it damages the structure. Users have no means to defend against such side-effects because until now it has been unusual to accept complaints about viruses, for example, on hot-lines.

Moreover, it follows from the principle of the function of viruses that they always interfere with the integrity of infected programs by their activity. This results from the fact that all ***viruses obtain control flow by theft***, that viruses steal control from the programs which they have infected. Usually, but not always, they steal control by modifying the code of the infected programs. An example of viruses which steal control without program modification, are companion viruses. ***By the theft of control the viruses act as parasites*** on the programs infected by them. This ability is given to each virus at the time of its origin. It is the inherently parasitic nature of the self-replication of computer viruses which interferes with the integrity of the programs affected by them.

Besides, by self-replication ***viruses waste computer resources***, particularly memory and processor time, which is also a form of doing harm. Although this form is often tolerated, it is ***unpredictable*** and in time-critical applications it can be substantial and is therefore intolerable.

It depends whether some of the given influences are demonstrated to be harmful ones. In all cases, by their ability to self-replicate and their parasitic nature, viruses violate the conditions of function of the programs affected by them, therefore the authors of the programs cannot guarantee the functionality of the programs, which restricts their author's rights.

From the above mentioned arguments, it is sufficient for everyday practice that the best which can be said about the simplest viruses, containing no code for harmful actions, is the following:

(1) The harmlessness of computer viruses is not guaranteed.

In other words, the usage of any virus is risky, because of the danger of violating computer activity. This riskiness of computer viruses results from their ability to parasitically self-replicate. Because without this ability the virus is not a virus, it follows that this riskiness is peculiar to all computer viruses, also in cases when they have no ability to execute harmful action, and even when they have the ability to execute useful action. It also follows from that, that the danger of harmful viruses does not rest only in their ability to execute harmful actions.

Useful viruses

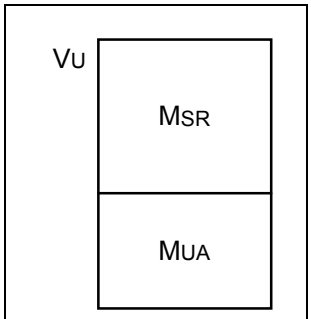

Fig.3 Useful virus

Generally a useful virus VU (Fig.3) consists of at least the following two modules: module of self-replication MSR and a module of useful action MUA. The useful action is usually executed by the virus on a certain condition.

A useful action which virus can execute is, for example, the compression of the code of an infected program, as is done by the virus {Cruncher} [Kaspersky-93] [Timson-93]. Another example is

the virus {AVV} [Kaspersky-94], which detects the presence of other viruses.

It is problematic to evaluate the virus VU as unambiguously useful, because it is unknown whether the usefulness of its action outweighs the riskiness of its self-replication. Moreover, it is problematic to compare the usefulness of the action to the riskiness of the self-replication. Even if the usefulness of the action was much greater, the riskiness of the self-replication, however low, might be intolerable, and therefore the virus as a whole could not be evaluated as unambiguously useful.

Let us consider a virus VC, whose useful action is a compression of the code of programs. Fig.4 shows the situation when N programs P_1 - P_N have been infected by the virus. Each of the programs P_i has been compressed at the time of its infection by the virus. Along with it a part of the virus VC acting as a parasite on it has been compressed. The rest of the virus, which is a decompression module D, has not been compressed and receives control at the time of activation of the program P_i . Module D decompresses the program P_i and the compressed part of the virus VC to their original state, control is passed to the decompressed remainder of the virus VC, and the rest of the process goes on as usual for viruses. This means that a program P_i infected by the virus VC behaves like a self-expanding program.

From the user's point of view, besides the above mentioned problems, there are the following interesting matters of fact. The compression module is present in each instance of the virus VC, in our case it is N-times, which is not the case in common compression programs. Next, it is interesting that the ***virus activity is uncontrollable***, because the virus itself finds the programs to be infected, fully ***autonomously***, according to the rules built into it. Due to this reason, the user is unable to decide on which programs the compression should be applied and on which ones it should not. Finally, it is interesting that ***viruses behave in an indeterminate way***, because their activity often depends on software configuration, sequence of executed operations and other parameters of random nature. Therefore it is generally ***unpredictable*** whether at a given moment the compression has been applied to any given program or whether it has been applied to all considered programs.

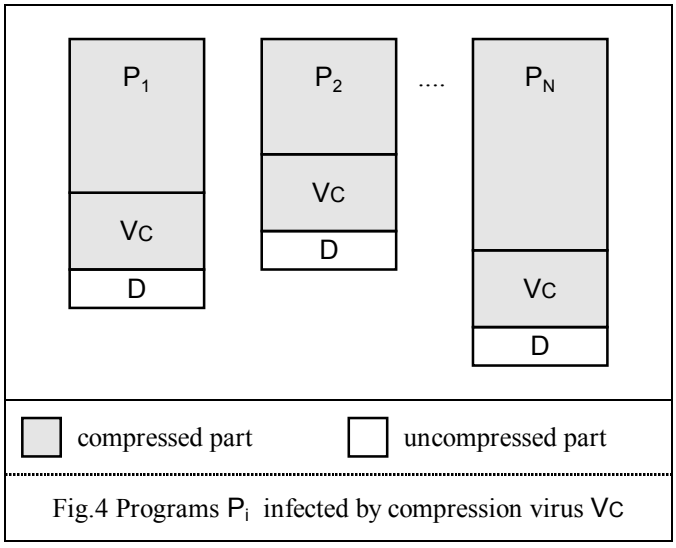

Fig.4 Programs P_i infected by compression virus VC

The above given facts disqualify the useful and harmless viruses to such a degree, that the least negative statement we can say about them, is the following:

(2) Harmless and useful viruses can hardly exist.

In the next section we will look at whether it is necessary to regret that useful viruses have such weak prospects.

Useful viruses vs useful non-viral programs

If a common, correctly implemented compression program is used, we avoid the problems inherent in compression virus VC. First of all, we avoid problems with uncontrollability and indeterminacy, because it is possible to state on which programs to apply the compression and,

after the compression is finished, it is apparent that compression has been applied only to the stated programs and not to others.

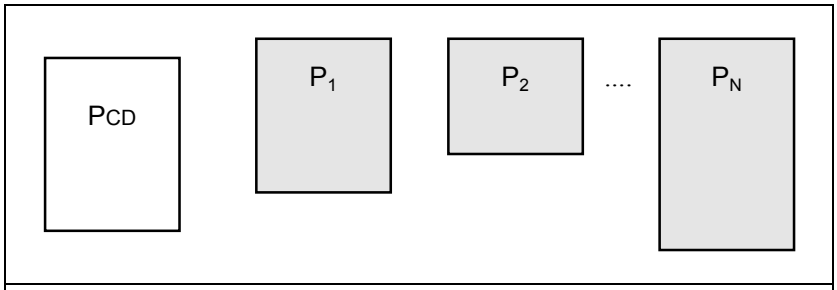

Fig.5 Programs P_i compressed and uncompressed by program PCD

The differences in the demands on memory and time are not negligible as well. The compression code together with the self-replicating one occur in each instance of the virus VC, that is, in each infected program (Fig.4). On the other hand, if we use a compression program PCD, which may or may not be memory resident, the compression code is necessary only in one instance and the self-replicating code is unnecessary (Fig.5). It can be seen, that the programs P_1 - P_N, on which the compression program PCD was applied, contain no extraneous code.

Similarly, the program PCD is more time-efficient, because it works on demand only, and not like the virus VC, which works every time it steals control. If the compression program PCD is memory resident, it works autonomously, which removes the last illusory advantage of the useful viruses, for which some of their proponents argue.

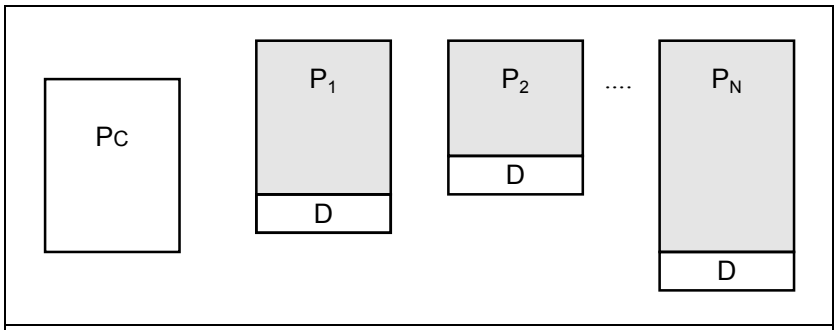

Fig.6 Self-expanding programs P_i compressed by program PC

In practice we also use compression program PC, which transforms the given programs into self-expanding form (Fig.6). A compression program PC compresses given programs P_1 - P_N and then adds to them a decompression module D, which automatically decompresses them at their activation. The addition of the decompression module influences the integrity of the given programs, but if the authors of the programs agree with this process, their author rights are not violated, and if they themselves apply such compression to their programs, the integrity of the programs is not affected.

The procedure, which was demonstrated in the comparison of compression viruses and compression programs, can be generalised in the following statement:

(3) For each virus which is able to execute an action, it is possible to implement a program, which is not a virus and which is able to execute the same action.

Therefore for each virus VU (Fig.3), which is able to execute a useful action using the module MUA, it is possible to work out a useful program PU (Fig.7), which is able to execute the same useful action as the virus VU. PU needs no self-replication code, therefore it is not a virus. Instead it contains a module of selective application MSA, by which the useful action is selectively applied according the commands of the user of the program PU. In an extreme, the program PU can contain a copy of the module MUA, which would guarantee the equivalency of an execution of the useful action. Using statement (3), the above given comparisons of features, shown for compression viruses and compression programs, is valid for every pair VU - PU, which executes the same action.

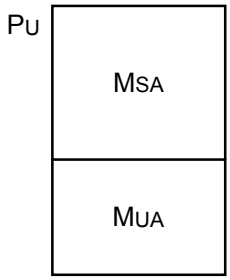

Fig.7 Useful program

Now we are at the end of the comparison of the features of the useful viruses VU and the useful programs PU. Their comparison overview is given in Table 1. From it and from statement (3) the following statement results:

(4) Useful viruses are useless.

This is so because useful programs are unambiguously more advantageous, as useful viruses have only one from the given list of positive features, which is the ability to execute useful actions. Otherwise the usage of viruses for useful purposes is ***hazardous***, because it is accompanied by several risks.

feature	useful virus VU	useful program PU
useful action	+ able to execute	+ able to execute
self-replication	– basic ability, without it virus is not a virus	+ does not need
parasitic ability	– basic ability, without it virus is not a virus	+ does not need
controllability	– autonomous, uncontrollable	+ user controllable
predictability	– indeterminate behaviour	+ predictable behaviour
memory usage	– unpredictably excessive	+ need not be excessive
processor usage	– unpredictably excessive	+ need not be excessive
	– *is risky and therefore negative feature*	+ *is positive feature*

Table 1 Comparison of basic features of useful viruses and useful non-viral programs

Conclusion

To justifiably speak about the existence of harmless computer viruses, it would be necessary to prove, or at least to show, how to implement them in such a way that it would be possible to guarantee their harmlessness. That would disprove the validity of statement (1) and at the same time open the possibility of the existence of unambiguously useful viruses.

To justifiably speak about the existence of useful computer viruses, it would be necessary to prove, or at least to show, that there exists an action which can be implemented in the framework of a virus and which cannot be implemented in the framework of any non-viral program. That would disprove the validity of statement (3) and therefore (4). Another possibility would be to prove, or at least to show, that there exists an action which is more effective to implement in the framework of a virus than in the framework of any non-viral program. That would disprove the validity of statement (4), but not (3). To justifiably speak about these viruses as being unambiguously useful, statement (1) may not be valid.

Until such proofs can be given, claims about the existence of harmless and useful viruses are but the products of ***wishful thinking*** of their proponents, and attempts to create and use them are hazardous. In the case of the useful viruses it is an unnecessary hazard, because useful non-viral programs do not carry the risks which viruses do. The hazardousness of the viruses results from the cumulative effect of risks connected with their usage. These are mainly the risks resulting from the parasitic self-replication of the viruses, from their uncontrollable and indeterminate activity, and from their unpredictably excessive usage of memory and processor.

Software engineering looks for programming techniques whose use minimises the risks of incorrect software function. This is why we consider as unsuitable those programming techniques, which the hazardousness of the computer viruses is based on. From the viewpoint of software engineering, ***viral programming techniques are dirty*** at least as unstructured or non-modular programming is, since their exploitation is dangerous.

It is known that all viruses in some way violate the integrity of the infected programs, which is a given due to their parasitic nature. This interferes with the author and user rights of the respective infected programs. The authors and users of those programs have the right to protection by law, to recompensation and to the prosecution of the culprits who spread viruses actively or support their spread through negligence.

References

[Cohen-85] Fred Cohen: Computer Viruses. Fred Cohen 1985.

[Cohen-92] Frederick B. Cohen: 'Wrong' Said Fred. Virus Bulletin, January 1992, 5-6.

[Kaspersky-93] Eugene Kaspersky: Cruncher - The First Beneficial Virus? Virus Bulletin, June 1993, 8-9.

[Kaspersky-94] Eugene Kaspersky: AVV - The Anti-Virus Virus. Virus Bulletin, January 1994, 10-11.

[Timson-93] Harriet Timson: Cruncher - Zipping or Zapping . Virus News International, May 1993, 31.

■

