Taming Lakatos' Monster -
Computer Virus Epidemics and Internet Security Policy

Viktor Mayer-Schönberger*
The John F. Kennedy School of Government / Harvard University

A few months back I received an email inviting me to speak at WebNet 2000. I was pleasantly surprised and ready to accept until I read the proposed topic of my talk "computer security and computer viruses". I was dumbfounded. I am a lawyer by training, teaching at a policy school and in my research I focus on information technology policy. Why on earth then do they want me to talk about computer security and computer viruses? I asked myself.

It may have been a computer error. Maybe two invitations got mixed up, and while I got one for computer security, a computer security expert might have gotten one to talk about cyberlaw and cyberpolicy. Maybe. In any event, it made me pause. I canceled my draft acceptance email, switched off the machine and pondered fate while biking home that evening.

And then it dawned on me. This wasn't a mistake after all. Far from it! Somebody must have remembered a previous life of mine, so to speak, and given me this wonderful opportunity to revisit a topic I once had been involved in so deeply. Thus this invitation to speak to you today provides me with an opportunity to revisit computer security, to uncover a bit of my own history, and to retest a few assumptions I made a decade ago. So I accepted – and here I am. Please join me for the ride as I recount a journey both professional and personal – of battling computer viruses, or - taming Lakatos' monster.

Spring 1991. Snow still covered the mountaintops surrounding Zell am See, my home town, a small Austrian village in the middle of the Alps. Five years earlier I had founded Ikarus

* Assistant Professor of Public Policy. I would like to thank Ed Lee for invaluable assistance in researching this paper.

Software. For three years then we had battled computer viruses. Our “Virus Utilities” were Austria’s best selling software. But the fight against computer viruses had gotten tougher. What used to be a trickle of a new virus every month had increased to a steady stream of a few new viruses every week, straining our limited resources and forcing us to issue updates much more frequently. I was worried that if the trend continued we would soon be incapable of keeping apace.

Then one day I came across an article dryly entitled “Levels of Generality in the Definition of Rights” in the fall 1990 issue of the University of Chicago Law Review.[1] I wasn’t particularly intrigued by the title – it was boring even by lawyers’ standards. But I knew its authors – Laurence Tribe and Michael Dorf. Tribe had been my Constitutional Law Professor and Michael my classmate in law school. So I thought I might have a look at what they had to say.

In the article Tribe and Dorf introduced me to the works of Hungarian mathematician and philosopher Imre Lakatos[2]. In his book “Proofs and Refutations”[3] Lakatos had suggested that human problem-solving, including physics and mathematics, is a process by which proofs become more rigorous as one subjects them to counterexamples and criticism. According to Lakatos the human problem-solving process happens in three distinct stages, which he called monster barring, exception-barring and lemma-incorporation.

In the first stage, monster barring, one attempts to negate a new problem by treating it as a monster, insoluble. If the problem persists, we reach the second stage, in which the problem is acknowledged, but incorporated into our knowledge as an exception to the established rule. Only in the third stage, the lemma-incorporation, will we accept the problem as such and modify our knowledge to incorporate a solution.

---

[1] Laurence H. Tribe & Michael C. Dorf, Levels of Generality in the Definition of Rights, 57 University of Chicago Law Review 1057 (1990).

[2] For an overview of his work, see Brendan Lavor, Lakatos : an Introduction (1998); see also Teun Koetsier, Lakatos' Philosophy of Mathematics : a Historical Approach (1991); Kostas Gavroglu, Yorgos Goudaroulis, Pantelis Nicolacopoulos (eds.), Imre Lakatos and Theories of Scientific Change (1989); Gunnar Andersson, Criticism and the history of science : Kuhn's, Lakatos's, and Feyrabend's Criticisms of Critical Rationalism (1994).

[3] Imre Lakatos, Proofs and Refutations: The Logic of Mathematical Discovery (1976).

Lakatos' model seemed to describe accurately in its first stage what I'd like to term the "computer virus scare". Computer viruses were seen as the insoluble monster, threatening the very foundations of our conception of information processing.

Pundits and self-proclaimed experts ascribed almost mythical qualities to computer viruses. Viruses were said to be

- invisible like the mythical Alberich wearing his magic hat,
- ubiquitous like the Greek deity Pan, and
- so plentiful as to cause a global information meltdown, evoking connotations to the biblical Armageddon.

This was a recipe for hysteria, not fertile ground for robust societal solutions to the computer virus problem. But that seemed not to matter. Computer viruses were dangerous, and writing about them was en vogue. The media was hungry for ever more shocking predictions on what havoc they would wreck, and the usual mix of consultants, industry watchers and spokespeople of anti-virus software companies willingly joined the scare.

The following story may illustrate this[4]: In late 1991 a new computer virus was discovered and named after the great Italian renaissance genius Michelangelo. It was constructed to destroy data on all infected computers at one day in early March of 1992. A European computer security expert promptly went on record proclaiming *tens of thousands* of PCs at risk.[5] An American anti-virus manufacturer, not to be outdone that easily by the Europeans quickly announced not tens of thousands but *millions* of PCs would be affected worldwide.[6] This caused an angry response from Europe, and the modified European prediction that *tens of millions* of PCs could potentially be affected.[7]

---

[4] Anne Branscomb has looked at an analogous story: the amount of damage caused by the Morris/Cornell worm; see Anne W. Branscomb, [article in a Rutgers Law Journal]

[5] [Brunnstein (VTC Hamburg)].

[6] [McAfee's non-profit arm].

[7] [Brunnstein (VTC Hamburg) again].

For anyone only mildly informed these predictions were ludicrous. The Michelangelo virus contained an odd and restrictive self-replication routine, thus severely limiting its potential impact.

Nevertheless, on the day Michelangelo was supposed to strike TV news crews scrambled to be present at the leading computer security companies phone support centers to witness the predicted global information meltdown. But the phones remained quiet.[8] Within minutes the media realized that this was a no-event and left, only to return a couple of years later when we experienced the Melissa virus scare.[9]

But on the whole we have now moved away from monster barring, Lakatos' first stage of problem solving during the last decade. We have learned to live with computer viruses, understand that sharing code exposes us to virus risks, and we use suitable precautions to limit the impact of a potential infection. To be sure, computer viruses are still seen as the exception, and when a wide-spread epidemic takes place, as with the "Iloveyou" virus this spring, people are still scared, taking extra precautions, clamping down on sharing files.

Similarly on the technical front, anti-virus researchers have been able to keep largely apace with the stream of new viruses. For once, the vast majority of computer viruses have never been seen "in the wild", in the real world of classrooms and office cubicles. Their spread has been limited to the digital petri dishes of virus creators and their counterparts in the anti-virus community.[10]

Secondly, computer security folks have successfully addressed the feared virus techniques of self-encryption and polymorphism.[11] To understand the evolution in computer virus

---

[8] Globally, there were some reports of Michelangelo incidents, but as Kephart et al remark "the number of machines infected by the Michelangelo virus was orders of magnitude less than one widely quoted estimate of five million, and the damage it caused was slight." See Jeffrey O. Kephart / Steve R. White / David M. Chess, Computers and Epidemiology, IEEE Spectrum, Vol 30, No 5, 20 (1993); Jeffrey O. Kephart / Steve R. White, Measuring and Modeling Computer Virus Prevalence, Proceedings of the 1999 IEEE Computer Society Symposium on Research in Scurity and Privacy, Oakland, California, May 24-25, 1993, 2.

[9] See Richard Ford, No Surprises in Melissa Land, Computers & Security 18 (1999), 300.

[10] [cites]

[11] See also Carey Nachenberg, Computer Virus-Coevolution, Communications of the ACM, January 1997, Vol. 40 No 1, 46.

technology over the last decade or so, a few words about computer viruses are, I think, useful.[12]

A computer virus is basically a self-replicating piece of software code. In most cases it also contains code which when triggered by some event causes an effect on the infected computer; from falling letters on the screen, to the creation of odd sounds, to more destructive effects like the erasing of the hard disk's partition and file allocation tables or the stepwise encryption of one's hard disk data. Virus researchers are not very interested in these effects. Their main goal is to understand the self-replication mechanisms.

Self-replication creates a second instance of the virus. In order to both avoid early detection and increase the chances of code execution, the computer virus is best off to add its self-replicated code to an already existing software piece. This is why such self-replicating programs are called viruses: they depend on a host piece of code to have their own code being executed.[13]

Current PC architectures offer two distinct kinds of such "pieces": boot sector on hard disks and floppy disks, and the code files, whether they are program files, or libraries or device drivers or something similar.

Any software code in the boot sector is read into memory at boot time and then executed. Adding virus code to the boot software is thus almost a guarantee for the virus code to be executed by the system. But the simple boot sector viruses' Achilles heel is their spread. They can only spread by copying themselves onto the boot sector of some removable storage medium, and – most importantly – by a PC being booted off that infected removable media. This is rare and happens if at all by accident, by – for example - leaving a floppy in the drive and rebooting.

[12] For more detailed descriptions, see […].

[13] Fred Cohen, Computer viruses (1987).

Viruses attached to files, on the other hand, can spread much easier.[14] They just need to copy themselves into program files and spread from one user to the next whenever these files are shared. Their disadvantage is that they have to "wait" until the infected host program is executed. Understandably therefore computer viruses attempt to infect system software pieces, code that is going to be run with a high degree of certainty right at start-up.

What can be done against viruses?

In a nutshell, anti-virus software searches program files and boot sectors for computer virus signatures and once found attempts – if possible – to extract the virus code from the host.[15] To evade such detection, some computer viruses have been constructed to encrypt most of their code with a changing key thus making detection harder.[16] But anti-virus experts discovered quickly that a code part responsible for decrypting the virus had to remain unencrypted and thus could be easily identified.[17]

A few virus authors then developed quite sophisticated algorithms, which would mutate the decryption code part with each generation. In principle, every generation was doing the same thing, i.e. decrypting the encrypted part of the virus code, but the actual steps were switched and altered, so that each virus had a different decryption routine. Anti-viral software looking for a specific signature would thus be fooled.[18]

This prompted a very sophisticated response from the anti-virus community – the virtual machine.[19] Modern anti-virus software basically runs each executable file in a virtual PC created in your PCs memory. The virtual machine will stop executing instructions after a while and the anti-viral software will analyze the results. If a mutating and encrypted virus

[14] They account for roughly 85 percent of computer viruses; see Jeffrey O. Kephart / Gregory B. Sorkin / David M. Chess / Steve R. White, Fighting Computer Viruses, Scientific American November 1997, http://www.sciam.com/1197issue/1197kephart.html.

[15] For more detailed descriptions, see […].

[16] [Cites]

[17] [Cites]

[18] [Cites]

[19] See Understanding and Managing Polymorphic Viruses, The Symantec Enterprise Papers, Volume XXX.

has infected a file, it will have decrypted itself by then and thus made itself visible to the anti-viral software. Creating a complete virtual PC is not an easy feat, but it ultimately has provided PC users all over the world with a very powerful virus-antidote.

This is basically where we are today. From a fundamental engineering perspective fairly little has changed since the first virtual machines were implemented in anti-viral software.[20]

In a recent paper, IBM anti-virus experts listed important loose ends, but none of them are fundamental threats to the basic anti-virus setup.[21] Let me give you two examples:

- Some computer viruses try to detect whether they are run in a virtual machine and will not execute their main code base if they think they are test-run by a viral antidote. For a computer virus it is actually quite easy to detect whether it is run in a virtual machine – yet, as most virtual machines do not completely emulate the entire PC. By testing for math coprocessor presence, MMX multimedia instructions or certain Ethernet chips, computer viruses may find clues as to whether they are run in a real or only partially completed virtual environment. The anti-viral communities' response of course is to complete the emulation in the virtual machines.

- Another potential problem is the time it takes to examine a typical user's hard disk for viruses. It is quite time-consuming if each program file has to be executed in a virtual machine and the result analyzed. Eight years ago one would stop the virtual machine after a couple of hundred individual programming instructions had been executed. Today computer viruses make extensive use of available operating system calls and thus integrate layers and layers of system code. In 2000 typical anti-viral software will examine

---

[20] Of course, sophisticated tools have been developed to aid the anti-virus experts in analyzing computer virus samples, and complex systems (including ones based on neural networks) have been built to identify unknown viruses based on behavior patterns; see Jieh-Sheng Lee / Jieh Hsiang / Po-Hao Tsang, A Generic Virus Detection Agent on the Internet, IEEE [???] (1997), 210; Jeffrey O. Kephart / Gregory B. Sorkin / David M. Chess / Steve R. White, Fighting Computer Viruses, Scientific American November 1997, http://www.sciam.com/1197issue/1197kephart.html; Understanding Heuristics: Symantec's Bloodhound Technology, Symantec White Paper Series Volume XXXIV.

[21] Steve R. White, Open Problems in Computer Virus Research, http://www.av.ibm.com.ScientificPapers/White/Problems/Problems.html

a program file by letting it execute a hefty 10 million instructions in the virtual machine before analyzing the results. And this takes time, even with gigahertz clock speeds.

However, the biggest concerns of anti-virus experts have not been the technical challenges, but the user perceptions. Microsoft's application software permits adding powerful macros to what used to be simple data files.[22] Similarly simple ASCII emails have now been replaced by emails containing executionable code.[23]

For the vast majority of unassuming users, data is still data and programs are still programs. They have yet to fully grasp the convergence of data and code into one unit, and to understand its consequences. This blissful ignorance – fostered - by the way - by the continuous reaffirmation of software producers that PCs can get more powerful *and* easier to use at the same time – this blissful ignorance has fueled the threat posed by both macro viruses and the recent "Iloveyou" bug.[24]

Macro viruses pretend to be valuable macros, embedded in Word documents, or Excel spreadsheets.[25] PC users may know that sharing a program may pose a computer virus risk, but most of them are conceptually unaware of the danger of sharing Word or Excel files.[26]

Similarly, the "Iloveyou" bug is a fairly simple visual basic program, which once started sends itself to all the people contained in one's email address book.[27] People start the self-

---

[22] [Cites to articles calling this a security risk]

[23] [Cites to MS "rich" emails]; Javascript and ActiveX pose related problems; see David Chess, The Future of Viruses on the Internet, http://www.av.ibm.com/ScientificPapers/Chess/Future.html; these techniques also pose vulnerabilities vis-à-vis Trojan Horses, see Sarah Gordon / David M. Chess, Where There's Smoke, There's Mirrors: The Truths about Trojan Horses on the Internet, http://www.av.ibm.com/InsideTheLab/Bookshelf/ScientificPapers/Gordon/Trojan/Trojan.html.

[24] See also Mark Kennedy, Script-Based Mobile Threats, Symantec AntiVirus Research Center, June 27, 2000.

[25] See also Paul Docherty / Peter Simpson, Macro Attacks: What Next After Melissa, Computers & Security 18 (1999) 391; Vesselin Bontchev, Macro virus identification problems, Computers & Security 17 (1998) 69.

[26] According to the ICSA Labs 6th Annual Computer Virus Prevalence Survey 2000, the vast majority of common computer viruses today are macro viruses; they account for 91 encounters per 1000 PCs per month (with similar VB script viruses accounting for another 22 encounters), compared with 2 encounters for file viruses and less than one encounter for boot viruses; ICSA Labs 6th Annual Computer Virus Prevalence Survey 2000, 17.

[27] VB script viruses account for 22 encounters per 1000 PCs per month; ICSA Labs 6th Annual Computer Virus Prevalence Survey 2000, 17.

replicating code by curiously double-clicking on the executable attachment of an infected email.[28]

If Microsoft had not made it so seamlessly to mix and combine data and code, these threats would probably not be as serious. However, as the problem here is one of human perception and understanding, anti-viral software can hardly provide a quick fix. Microsoft could, but is unable or unwilling to take a step back from combing data and code.

Another recent issue with computer viruses is constant broadband connectivity of many PCs to the Internet. A computer virus thus can access the net without many a user of infected PCs noticing.

In my personal assessment, this and the seamless mix of data and code, both linked to false user perceptions of separated-ness, pose substantially heavier threats to computer security than any increased technical sophistication of computer viruses. It seems that after more than one decade of end user education the single most vulnerable part still is – the user. Unfortunately this is also the hardest part to change.

If, however, our system of protection has such an inherent weak link, why then - one may ask – have computer viruses not caused the predicted informational Armageddon yet? Indeed, computer virus authors must certainly have been well aware of the human factor and hard at work exploiting it. Why then haven't all of our computers been infected and all of our hard disks been erased?

Not only conventional wisdom would suggest that. In the late 1980s / early 1990s some experts looked at epidemiological patterns and predicted a rapid increase of infected computers worldwide, bringing havoc to the fledging information economy.[29] Despite some high profile incidents like Michelangelo, Melissa and Iloveyou this did not happen. Why?

---

[28] Email attachments are responsible for 87 (!) percent of all computer virus encounters in 2000, up from 56 percent in 1999 and 32 percent in 1998; ICSA Labs 6th Annual Computer Virus Prevalence Survey 2000, 32.

[29] W.H. Murray, The application of epidemiology to computer viruses, Computer Security 7 (1988) 139.

Researchers at IBM's Antiviral laboratory have given a very plausible answer.[30] They, too, looked at epidemiology. In principle, epidemiological models are centered around a fairly simple threshold condition. This epidemic threshold occurs when the rate of contacts between two viral hosts (two PCs) is higher that the rate of hosts being cured from the viral disease.[31] Such a case will then lead to a drastic increase in infections.

IBM's experts, however, added two important modifications to this basic setup. The basic model is based on an even spread of the infection through transmission. But in practice we do not share programs with anyone and everyone. Instead, we mostly stick to our co-workers, friends and acquaintances. This restricts transmission flows and paths. Adding this uneven topology of transmission networks to the model makes the spread of the infection much less likely.[32]

Secondly, IBM's experts looked at how we combat computer viruses. They observed that global media attention drawn to the outbreak of a computer virus infection, coupled with the user's own communication channels and networks makes it likely that the elimination of the computer virus happens in a much faster and more coordinated manner than one would expect. This fast spread of "kill signals" further dampens the impact of an epidemic spread.[33]

---

[30] Jeffrey O. Kephart / Steve R. White / David M. Chess, Computers and Epidemiology, IEEE Spectrum, Vol 30, No 5, 20 (1993); see also Jeffrey O. Kephart / Steve R. White, Measuring and Modeling Computer Virus Prevalence, Proceedings of the 1999 IEEE Computer Society Symposium on Research in Scurity and Privacy, Oakland, California, May 24-25, 1993, 2; other models have been suggested as well, see B.C. Soh / T.S. Dillon / P. County, Quantitative risk assessment of computer virus attacks on computer networks, Computer Networks & ISDN Systems 27 (1995) 1447.

[31] [IBM paper], Ingemar Nåsell, The threshold concept in deterministic and stochastic models, in Denis Mollison (ed.), Epidemic Models: Their Structure and Relation to Data (1995), 71.

[32] Jeffrey O. Kephart / Steve R. White, Measuring and Modeling Computer Virus Prevalence, Proceedings of the 1999 IEEE Computer Society Symposium on Research in Scurity and Privacy, Oakland, California, May 24-25, 1993, 2; see also Jeffrey O. Kephart / Gregory B. Sorkin / David M. Chess / Steve R. White, Fighting Computer Viruses, Scientific American November 1997, http://www.sciam.com/1197issue/1197kephart.html; Jeffrey O. Kephart / Steve R. White / David M. Chess, Computers and Epidemiology, IEEE Spectrum, Vol 30, No 5, 20 (1993).

[33] Jeffrey O. Kephart / Steve R. White, Measuring and Modeling Computer Virus Prevalence, Proceedings of the 1999 IEEE Computer Society Symposium on Research in Scurity and Privacy, Oakland, California, May 24-25, 1993, 2; see also Jeffrey O. Kephart / Steve R. White / David M. Chess, Computers and Epidemiology, IEEE Spectrum, Vol 30, No 5, 20 (1993).

Taken together, these two factors help explain what we have witnessed in reality: computer virus epidemics still occur, but the equilibrium between the spread of the infection and its containment that follows a short, quick peak happens at a much, much lower level than originally anticipated. In essence, computer virus infections peak, and then the number of infected computers decreases fairly rapidly to find equilibrium at a very low range or f.e. through technological innovation may even drop to zero.

Such voices of reason, of solid scientific analysis are a reassuring sign that we may be on the right track to move from Lakatos' second to the final third stage.

But much work remains to be done. IBM's epidemic models, while quite sophisticated, still lack some polish. The uneven network topology has only been incorporated in the model in a limited way, too much emphasis has been placed on spread, too little on establishment and persistence, the first and last of the three main epidemic stages.[34] Furthermore, IBM's model is based on a deterministic view of transmission and spread, not a stochastic one, although such stochastic models may be better suited for understanding and predicting computer virus epidemics.[35]

There is, however, another aspect of computer virus epidemiology which has been largely neglected so far. Existing data on the overall spread of computer viruses shows not only around 95 percent of all known viruses have never been seen "in the wild".[36] They also indicate that the most common viruses are not the most dangerous ones. Indeed, it seems like among the most common computer viruses are many fairly benign ones, which have neither the highest damage potential nor the most efficient infection mechanism. The current static epidemiological models fail to easily account for this observation.

---

[34] See generally Denis Mollison, The Structure of Epidemic Models, in Denis Mollison (ed.), Epidemic Models: Their Structure and Relation to Data (1995), 17.

[35] See Ingemar Nåsell, The threshold concept in deterministic and stochastic models, in Denis Mollison (ed.), Epidemic Models: Their Structure and Relation to Data (1995), 71.

[36]

The reason may be simpler than one would think. An important part of containing a virus epidemic is the determination of the affected system, whether it is the immune system or society at large to get it under control. A deadly infection will push the system to relatively drastic measures, while a comparatively benign infection may only cause some haphazard reaction. It is, in a sense, like the common cold. Its widespread presence may not only be caused by the constantly changing shape of the virus but also by the human immune system's lack of necessity to really combat it.

Similarly, as epidemic data demonstrates, computer users around the world are particularly vigilant when informed of an impeding or ongoing severe computer virus threat. In such cases the number of computer viruses detected increases drastically, as many users start to scan their systems for viruses. Many of the viruses they find, however, are not of the strand the computer users have been warned about. Their sudden vigilance – scanning their PCs - turned up not so much the feared strain than more benign viruses, which have been residing in many PCs for quite a while. And without the scare these benign viruses would have remained undetected.[37] If I were a computer virus author I would write a pretty benign, non-obtrusive virus. It's the best long-term survival strategy.

There is more to this observation. If we – as I suggest - take the human reaction to computer viruses as related to the danger posed by a particular virus, we have to conceive of the entire population of computer users and their more or less vulnerable PCs as a dynamic system. This system then reacts as a whole to the relative threat it experiences. It is my impression that the epidemiological data can best be explained by taking such a systemic view. Whenever there is a massive threat the entire societal system reacts: media report on it, anti-viral experts are hard at work for timely cures, network specialists install scanning firewalls, MIS departments issue warnings and clamp down on free program sharing and an increased number of users install anti-viral software. It is, if you want, like the empire striking back. But this time, the empire is not an uncontrollable mammoth. Its internal dynamics permit it to quickly adapt to the changing threat situation.

[37] See also Jeffrey O. Kephart / Steve R. White / David M. Chess, Computers and Epidemiology, IEEE Spectrum, Vol 30, No 5, 20 (1993).

Some experts have suggested an immune system metaphor to describe this systemic reaction.[38] I think this is extending the "virus" connotation a bit beyond its breaking point. Rather I would like to suggest we might gain further insights into and understanding of the behavior of this system by looking at Luhmann's system theory and its description of auto-poesis.[39] The global network of computer users with all their intricate formal and informal communication channels looks to me quite like the auto-poetic system Luhmann analyzes. By looking at these multiple information channels, their feedback mechanisms and how they are connected in a system-wide network leading to concrete action we might gain not only a better understanding of what is happening but also additional insights about how to maximize the efficiency and robustness of our "system".

Granted, practitioners care about solutions, not appropriate metaphors or theoretical models. But they must not overlook that we are still in Lakatos' second stage. Yes, despite a few lapses into monster barring, we generally have moved from stage one to stage two. We have understood that computer viruses are exceptions to the well being of our system. And in such exceptional cases of a computer virus epidemic our system takes measures, which limit the impact. In this second stage, we learned to live with computer viruses in a kind of uneasy truce, acknowledging the danger, taking some precautions and fearing potential outbreaks. But we have not yet solved the problem, we have not yet – in Lakatos-speak - "lemma-incorporated" it.

Some have argued that we will never reach this third stage.[40] We will never be able to deal with infections like our immune system does. These skeptics reason that such are the limitations of the computer itself, the Turing-machine model of information processing. The skeptics may be right fundamentally. As we continue to run software on our machines, which we have not completely written ourselves, we remain vulnerable to outside attacks and

---

[38] Stephanie Forrest / Steven A. Hofmeyr / Anil Somayaji, Computer Immunology, Communications of the ACM, October 1997, Vol. 40 No 1, 88.

[39] Niklas Luhmann, Social Systems (1995).

[40] F. Cohen, Computer Viruses, Theory and Experiments, Computers & Security 6 (1987) 22; Thimbleby et al have suggest that this claim is not sustainable, as the Turing machine is an ideal machine; see Harold Thimbleby / Stuart Anderson / Paul Cairns, A Framework for Modelling Trojans and Computer Virus Infection, Computer Journal, Vol 41 No. 7 (1998), 444.

can never reach a level of absolute security. But in their zeal they overlook that our level of analysis has shifted from the technical code level to the systemic level.

And this shift of perspectives may enable us to advance to Lakatos' third step, to solve the computer virus problem on that systemic level. IBM's anti-viral experts have suggested that all PC users (or at least all users of its anti-viral software) should become part of a global immune system.[41] This systems approach attempts to leverage the Internet as a communication and signaling network, quickly alerting others of possible infections and transferring samples of infected code electronically to the anti-virus research lab. In suggesting what they term an "immune system", these experts hope to overcome one of the painful Achilles heals of the current setup. Computer viruses may spread quickly by using the Internet, while there is not yet an established, automatic mechanism by which infected samples and virus cures are transmitted over the network almost instantaneously. Overcoming this would certainly be a huge step forward. Both IBM and Symantec have announced concrete plans to establish such a network, and other large anti-viral companies are following suit.[42]

Squeezing out inefficiencies in the transmission of samples and the distribution of cures is a very important qualitative step to improve the systemic response. But as anticipated, this setup also has its limitations: samples will only be transmitted to IBM and Symantec. We, the millions of computer users around the world have to trust that their capacities and capabilities are sufficient to promptly respond even to the most sophisticated computer virus

---

[41] Jeffrey O. Kephart / Gregory B. Sorkin / Morton Swimmer / Steve R. White, Blueprint for a Computer Immune System, http://www.av.ibm.com/ScientificPapers/Kephart/VB97/index.html; Gary Taubes, An Immune System for Cyberspace, http://www.research.ibm.com/resources/magazine…/immune496.htm; See also Sara Hedberg, Combating computer viruses: IBM's new Computer Immune System, IEEE Parallel & Distributed Technology, Summer 1996, 9; Jeffrey O. Kephart / Gregory B. Sorkin / David M. Chess / Steve R. White, Fighting Computer Viruses, Scientific American November 1997, http://www.sciam.com/1197issue/1197kephart.html; this "systemic" approach differs from "immune system" approaches focused on singular systems; for the latter (which may be combined with a systemic approach) see Patrick D'haeseleer, An Immunological Approach to Change Detection: Theoretical Results, IEEE [???] (1996), 18; Patrik D'haeseleer / Stephanie Forrest / Paul Helman, An Immunological Approach to Change Detection: Algorithms, Analysis and Implications, IEEE [???] (1996), 110.

[42] For research in this area see Takeshi Okamoto / Yoshiteru Ishida, A Dsitributed Approach to Computer Virus Detection and Neutralization by Autonmous and heterogeneous Agents, IEEE [????] (1999) 328; Robert E. Marmelstein / David A. Van Veldhuizen / Gary B. Lamont, A Distributed Architecture for an Adaptive Computer Virus Immune System, IEEE [????] (1998), 3838.

attack. We as a system have to trust a small part of us, experts in one anti-virus lab to “save” the world.

I have difficulties to understand why we should do that, why we should place all of our eggs in one nest, and not do, what we would in any other circumstance: hedge our bets, emphasis collaboration, information, communication, robust debate, open criticism and thus continuous improvements.

And finally this gets me back to my own personal story and gets us back to 1991.

So Tribe’s and Dorf’s article introduced me to Lakatos, and made me start search for the “lemma incorporation”. I reckoned that the main deficiency of our anti-viral endeavors had not been lack of technology, but lack of cooperation. Many dozens of anti-virus companies and research organizations were collecting virus samples, meticulously dissecting them, then incorporating this raw data into software updates. Each one of these firms closely guarded their virus samples and their virus analysis. This was, they felt, their core intellectual property, the very chance to gain a competitive advantage vis-à-vis other anti-viral products.

In their individual drives to increase their market share by keeping the virus information secret, they oversaw that with each little of their gain we all were loosing. We all were loosing because virus information was not shared among anti-viral experts. Instead it remained in little pockets in the various companies and research labs, insulated from criticism, which may expose flaws or foster improvements and out of reach for everyone except each company’s very own customers. If we could not solve this structural flaw of the system, I thought, we as a system would never be able to overcome being disadvantaged in fighting computer virus infections.

My goal was to overcome this structural problem, and thus to unleash the power of human cooperation within a global network. But how?

The answer was quite simple: open up, and share virus information – among researchers, anti-viral companies, and customers throughout the world. Of course, one needed to address

the security issue of how to share virus information without actually sharing the virus itself. And how to make sure that information is unambiguous and standardized, and thus can easily be understood and put to the test by others.

Rumor had it in 1991 that a well-known competitor at that time, Solomon Enterprises, had started to develop a formal computer language to program virus removal. It was – so typical - for internal use only, intended to speed the update process. Their secretive thinking did not permit them to understand that this could be used quite differently. And they had no intentions to let others in on their development.

So in May 1991 I wrote a memo, outlining my idea of global anti-virus cooperation, of opening our heavily guarded corporate information treasures and shift the momentum towards "lemma incorporation". The plan was as bold as it was simple.

All virus researchers, all anti-virus companies would use the same standardized way of describing computer virus qualities and behaviors, from how to detect them to how to remove them. This virus information could be shared without risk of causing infections. And by keeping it quite formal and standardized, it could be incorporated into anti-viral software automatically. It was like compiling a piece of code and linking it to the main project. Sharing virus information in such a way made it possible for virus researcher to divide up their work, and to evaluate each other's results much more easily. A virus sample found in Asia could be analyzed right there and through the speed of the net shared with everyone else. I even envisioned end users to be able to download the latest such virus information updates from trusted servers and incorporate it in their anti-virus scanners regardless of its vendor. For example, IBM's virus information could be incorporated and "understood" by McAfee's scanner. Virus information once standardized can be shared, tested and applied without reliance on a particular producer. This, I was convinced, would truly unleash the power of the network, the power of global cooperation, enabling us to enter Lakatos' famed third stage – at least on a system level.

To proof my point, we created a Virus Description Language (ViDL) and a compiler of sorts to add ViDL information to our virus software. We then developed tools to automatically

analyze viruses and generate parts of the ViDL description. We even created a database engine to manage the many individual records of ViDL descriptions. Almost to the day nine years ago, I demonstrated how in a matter of seconds a readable ViDL description of a new virus could be incorporated into an off-the-shelf virus scanner. And then, at that day, I put ViDL and its virus information content in the public domain.

My idea was that anti-virus companies could still compete against each other in terms of speed and ease of use. The one quality, however, where cooperation should replace competition was computer virus analysis. This is what I felt was right. This is what we deserved as users, and what would make us advance as a whole.

A few months later ICSA, the International Computer Security Association, a respected Washington-based independent tester of anti-viral software, announced that it would standardize on ViDL.[43] By March of 1992 I had calls of interest from all the major vendors. A month later due to a tragedy I had to leave my company. But ViDL, I thought, would live on and its momentum would continue.

Was I wrong! Today, almost a decade later, there is, as I have mentioned, renewed talk within the anti-virus community of creating a global "immune system". But this "immune system" is vendor-specific. And anti-virus companies still consider computer virus information their core intellectual property, to be hoarded, to be locked up, and never to be shared. In 1993 Solomon Enterprises won a Queen's award for their formal virus language[44], but they never released it to the general public. Today computer virus companies opt to rather leave their own users without a cure than to cooperate. It seems as if the entire Open Source movement[45] of the last decade, the momentum, which gave us among many others Linux, Mozilla and Darwin has not happened for the anti-viral community. Forgive me for

[43] Viktor Mayer-Schönberger / David Stang, Introduction to ViDL, a Virus Description Language, Virus News and Reviews 248 (1992).

[44] [Cite]

[45] See only Chris DiBona / Sam Ockman / Mark Stone, Open Sources – Voices from the Open Source Revolution (1999); Eric S. Raymond, The Cathedral & the Bazaar, Musings on Linux and Open Source by an Accidental Revolutionary (1999).

my rhetoric, but if cooperation could offer a faster, more effective cure, should we all continue to suffer, to be kept hostage by a precious few?

I could feel vindicated by history. Permitting others to look at one's code, one's treasured intellectual property gems is now in fashion. But this is not about personal vindication. This is about computer viruses, the threat they pose and how we best can combat them. This battle we all have lost - so far.

So far. So let me finish my talk today by urging you all to think about the computer virus threat not as a given, but as something we may overcome - of sorts; to think about it in terms of all of us and our share in pushing for a solution; to call on the computer virus experts, the anti-virus companies to take seriously a lesson from Open Source, to cooperate for real cures, and thus to unleash the very power we have to combat the viruses – the *network of us*.