Reply to 'Comment on "A Framework for Modelling Trojans and Computer Virus Infection" ' by E. Mäkinen

FRED COHEN

Sandia National Laboratories
The University of New Haven, New Haven, CT
Email: fc@all.net

Received 3 April 2001

There may be some relatively interesting things to do in developing Turing machine models of operating systems. The original Turing machine model of operating systems used to model viruses (Cohen [1]) does this to a limited extent and can also be used as a model of networks, but it fails to capture the full richness of modern operating systems in detail. The advantage of this is generality, but the disadvantage is the inability to specifically model the detailed events in current operating systems.

There are computer viruses that operate in nonuniversal Turing machine environments—for example, there is a set of UDP viruses that operate in the UDP Internet Protocol environment which is not Turing capable as far as I can tell. For this reason, we should not restrict our study to Universal Turing Machines.

Mäkinen [2] leaves a great deal of detail out that might make for a new perspective on this theme. For example, how do you 'find a description of some other Turing machine . . . on a tape'? This seems to me to be undecidable.

To the extent that the problem is that Turing machines are not very useful for doing detailed models of most modern computer systems, I have to agree—I think I even said so in my original work in this area. The notion of viral sets that operate in certain TMs is critical to understanding viruses. The use of histories, while it may be painful to understand at first glance, was the only way I could find to express what I wanted to express. These constructs have served us well in properly encompassing all of the viruses that have come along since and are likely to come along henceforth. They also do a very good job of dealing with issues of evolution, co-evolution, etc. in computer viruses, which was why I chose these constructs in the first place.

Mäkinen was essentially taking issue with Thimbleby *et al.* [3]. Thimbleby *et al.* certainly seem to draw many questionable conclusions without substantial support. For example, the original Turing machine model from 1986 does an excellent job of modeling things like viruses encrypting subsequent versions of themselves—and in fact, the entire class of evolutionary mechanisms. That is why the 'histories' mechanism Thimbleby *et al.* [3] complain about was included. The masquerade problem is also covered by the model of a timesharing system in this dissertation—in fact, the Trojan in the path variant was specifically covered in this work because that is precisely how I came to understand viruses the first time. So I strongly disagree with the notion that Turing machine models in general cannot cover these issues, and specifically question whether the authors have read and understood the theoretical work in this area.

In terms of Turing machines being infinite and real computers being finite, of course this is essentially true. But the real question is whether the finiteness of real computers makes much of a difference. For Turing machines, problems are 'undecidable', while for real machines, they are simply too complex to be solved by any real-world solution mechanism. The problem with a 'theoretical framework' for 'real computers' is that it is, in some sense, an oxymoron. Theoretical frameworks and models are designed to abstract the critical elements of real systems for the purpose at hand so that we can use reason and mathematics about those elements while ignoring things we consider irrelevant for our purposes. Real operating systems run on real computers, and real computers are subject to all sorts of physical issues, like solar flares and power failures, that might impact a lot of theoretical results if we were forced to include them. The question is how far to go in modeling the 'real world' and the answer depends on what you are trying to accomplish with your models.

If Section 2 of Thimbleby *et al.* [3] is questionable, Section 3 of this same paper is thought-provoking. I do think that, if we looked deeply into it, we might find that the definition of a virus in Section 3 would include many things we don't really want to include in this class, including

all operating systems and applications that 'happen to be infected'. In other words, they seem to really be saying that we should call any infected program a virus. This fails to differentiate a virus from the rest of the code in a program, and this creates untold havoc in making sense of the virus and antivirus issue. It is sort of like saying that you are a microphage because you contain a microphage—but of course you are a person.

It also appears that Thimbleby *et al.* [3] tried to use formal proof methods on the informal definition of viruses rather than referring to the more theoretical work on this subject (their citation 28) or the more definitive book *Computer Viruses* (Cohen [1]) which they apparently did not find in their research (as opposed to the paper by the same name found in *Computers and Security*, Cohen [4]).

This is no doubt why they find problems with the informal methods of showing undecidability. Thimbleby *et al.* [3] are using a theoretical framework to evaluate a nontheoretical result and ignoring the theoretical result that solved the precise problems they find with the nontheoretical result—and did so more than 15 years ago.

In Section 4, Thimbleby *et al.* [3] express possible uses for their framework, but they give no examples of these uses or practical methods for going from the framework to those solutions. In order to make such a paper really worthwhile, the authors should come up with new results. For example, they could tell us about a new easily computable way to detect viruses that covers a large (infinite) otherwise noncovered class of viruses. This would show that their theoretical framework has real power and value (anything that detects less than an infinite class is trivial relative to the infinite size of the class of viruses, but if we carve away enough of the infinite subclasses, we may largely cover the space of viruses even while leaving an infinite but very hard to implement class of indetectable viruses left).

I guess I have revealed myself on this matter and all matters of theory without utility. I am probably therefore unfit for and certainly not very interested in the deep exposition of mathematical truths without utility or novelty, and I revel in that lack of fitness and interest.

REFERENCES

[1] Cohen, F. (1985) *Computer Viruses*. ASP Press.

[2] Mäkinen, E. (2001) Comment on 'A framework for modelling Trojans and computer virus infection'. *Comp. J.*, **44**, 321–323.

[3] Thimbleby, H. W., Anderson, S. O. and Cairns, P. (1998) A framework for modelling Trojans and computer virus infection. *Comp. J.*, **41**, 444–458.

[4] Cohen, F. (1987) Computer viruses—theory and experiments. *Comp. Security*, **6**, 22–35.