

## Can Viruses Be Detected?

*Jennifer Lapell* 2000-06-17

Think Ebola

Think Ebola. That's what happened to Denis Tumpic. Not the real Ebola virus, but its catastrophic twin in the computer world. He turned on his computer one morning, and it refused to boot. At first, he thought the message that came up was a joke: "Your Amiga is alive." It was no joke, though. "What's that line from Jurassic Park?" he muses. "Life finds a way."

Like the fabled sorcerer's apprentice, a virus had been eating away bit by bit at Tumpic's system, corrupting his data behind his back. It took weeks before it cleared its throat and roared to get his attention. And by the time he noticed that anything was wrong, his operating system, programs and data were corrupted beyond recognition.

In real life, we fear the Ebola virus and its relatives not just because they are so deadly but because they invade our bodies in secret and incubate silently within us. If there were some way to find them before it was too late, we like to believe that a cure might even be possible. In fact, with diseases like cancer, early detection often does mean a better outcome. Imagine if that was possible with computer viruses as well.

Tumpic, a Toronto-based software developer, lived to tell the tale; it was just a computer virus, after all. He still does tell the tale, even now, ten years later, because the feeling of betrayal when your system is invaded never fades away completely. In fact, the attack on his Amiga system is one of the first things he mentions when he talks about Fred Cohen's work on computer viruses. For Tumpic and others who deal with computer security issues every day, Cohen's lessons are pivotal.

Known as the great-granddaddy of the computer virus world, Cohen modestly points out that he didn't actually coin the term "virus" or write the first experimental ones back in the 50's and 60's. Cohen was the one who clarified the meaning of the word, though. Back in 1987, he defined a virus as "a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself." There has been some bickering over fine points of Cohen's definition, but to this day, it's still widely accepted as capturing all the truths of what a computer virus is.

## Why is the Problem Getting Worse?

We've known about viruses since the 1950's. They've been under the microscope ever since then. The obvious question after so many years is "Why haven't they got this problem licked by now?"

In 1990, Cohen wrote that there were "over 100 well-known real-world computer viruses." Consider also that today, according to some estimates, there are over 50,000 viruses. Given that, it's not enough to say that the problem is not going away. As PCs proliferate and the potential for damage from viruses grows exponentially, the urgent question really must be, "Why is the problem getting worse?"

Your systems administrator may not know the answer, and your anti-virus software vendor isn't saying. We have to look to Cohen again for that answer. Cohen didn't just define computer viruses. He also earned a Ph.D. proving that it was impossible to create an accurate virus-checking program. Just as there has been some bickering over his definition of a virus, experts have argued back and forth since then about the details of Cohen's Theorem. None so far have proven him wrong. His proof is so diabolically straightforward that contradicting it comes down to arguing that "True" equals "False."

Cohen's discussion rests on work done in the 1930's by Goedel and other philosophers in the field of logic. Goedel used his Incompleteness Theorem to demonstrate there will always be complexities which cannot be handled by mathematics, logic, or any other "language" defined by our limited imaginations. Alan Turing and other mathematicians quickly realized that those "languages" include computer languages.

These thinkers posed a question which has had an irreversible impact on theoretical computer science: "Is it possible to write a program to determine if any given program works properly?" Their answer, surprisingly enough, was "No." Though this problem looks easy enough, they managed to prove that no program could ever exist which analyzes program behaviour. Now fast forward to the 1980's, when computer users were beginning to realize that the "virus issue" was a bigger problem than they had ever anticipated. This is where Fred Cohen comes in.

Cohen picked up the ball with his Ph.D. thesis in 1986, and extended Goedel's theorem to prove that no perfect virus checker can ever exist. His theorem is devilishly simple, relying on a

technique known as "proof by contradiction."

Here's how it works: Imagine for a second that Cohen is wrong, and that you have written a perfect virus-checking program to prove it. Let's call it "PVC" and think of it as a black box. You feed any program into PVC and out pops an answer: "Yes" or "No." Got your black box ready? Good.

Now that you've got your program, I also write a little program. I'm going to call it Ornerly Program (OP), and it has just two lines:

OP: 1. Ask PVC if I, OP, am a virus. 2. Follow one of these two possible responses: \* If PVC says "No" (I'm not a virus), then (infect) TAKE OVER THE WORLD! \* If PVC says "Yes" (I am a virus), then (don't infect) do nothing.

Remember the old riddle about the barber who shaves every man who doesn't shave himself? The paradox there is that if the barber shaves himself, he cannot shave himself, for he only shaves those who don't shave themselves. Because he can't shave himself, though, he is a man who doesn't shave himself ... which means that he must shave himself.

Ornerly Program is the computer equivalent of that barber. If PVC, which is perfect, decides that OP is a virus, then it isn't a virus. OP can only act as a virus if PVC determines that it is not a virus, just as our barber can only shave himself if he resolves to grow a beard. And so we find ourselves, in both cases, going around and around in circles.

A paradox like this can't exist in the real world, and so our underlying assumption -- that the Perfect Virus Checker program can exist -- must be inherently flawed. The assumption creates a paradox situation, and so the PVC "black box" -- a perfect virus-checking program -- not only does not exist, but cannot exist. This is known as Cohen's Theorem, and it has remained basically unchallenged to this day.

Real-World "Virus Checkers"

If Cohen has indeed proved that a virus checker is theoretically impossible, what about the little applet running in your system tray, or that virus-list update you just downloaded last week? What are they doing, if not checking for viruses?

Actually, they're not true virus checkers. There are two theoretical models for a virus-checking program: it can either look at the activity of the program or it can examine the virus as a simple bitstream. Cohen's Theorem assumes the former, that the checker analyzes program behaviour. Every virus-checking program on the market, though, takes the second approach and analyzes the form of a program, not its purpose.

Some virus checkers will attempt to monitor "virus-like" activity, such as writing to hard disk boot sectors, but in general, their entire strength rests on the bitstream approach. Your "virus-checking" program looks through programs before they are run to find suspicious sequences of bits known as "virus signatures." Virus checkers don't face Cohen's theorem head-on at all; they dodge the problem altogether and in doing so, Cohen suggests, they may do more harm than good.

Current virus checkers assume that a virus is just a simple sequence of bits; ones and zeroes in a known order. With that model, finding a virus is easy. Just mechanically compare bits and alert the user when a virus signature is found. Locating a needle in a haystack is not so difficult if you can sift the hay to filter it out. You could even think back to experiments in grade school and fish around for it with a magnet.

Imagine now that the needle doesn't look like a needle anymore. Its size or shape has changed, and filtering might not work. Maybe its fundamental properties are the same: your magnet might still do the trick. But what if the needle could mutate and change its properties so it was no longer made of metal at all? The magnet wouldn't work either. In fact, you wouldn't know what you were looking for, or what to do when you found it.

The real reason virus checkers don't work well is because they are all based on that flawed assumption, that a virus will always look and act the same. They pretend that the needle will always be a needle.

In real life, though, computer viruses can mutate well beyond recognition. They modify their own signatures and spread in almost unlimited variations. A truly effective virus checker would have to look for every permutation of the virus signature, and while the possibilities might not be infinite, the problem is too large for a real-world computer to tackle.

The vendor who confidently sold you a virus-checking solution will probably never admit that they don't know what to look for. But, as Cohen proved, they can't really search for viruses, so

they settle for scanning bitstreams for known viruses. That's why you're always downloading those "DAT" files to update your software with new virus signatures.

The danger of this approach is that those same vendors will proudly assure you that you have the best protection available. What they don't mention is that you still have little or no protection against viruses that have not been created yet. Addressing this false sense of security, Cohen is clearly a man with a mission: "[these products] can only detect things they know about ahead of time. It is a fundamental flaw in the technology that the public has chosen to buy."

## Real-World Solutions

Surprisingly, though his name has been synonymous with computer security for the last two decades, Cohen publicly brags that he does not use virus-checking programs himself. The bigger surprise may be that he has "not been successfully attacked yet."

Cohen's Theorem proves that we can never truly find out if our system is harbouring a virus. So how, according to Cohen, can a typical user protect his or her system from disaster?

His answer is simple. For typical computer users, though, it may be harder to swallow than the sting of the virus itself.

Computer viruses proliferate, Cohen writes, because we embrace the very technologies that make them possible. In a recent online discussion of the LoveLetter virus, Cohen wrote: "We have a tradeoff between convenience and security, and in order to get the general population to use computers, convenience is the key to success."

LoveLetter and its relatives have been so successful because consumers demand convenience and Microsoft and other vendors are happy to comply. Consumers "need" e-mail software that will open and run attachments. If we believe industry trends, we "need" word processing software that runs macros, we "need" applications (and operating systems) that update themselves automatically over the Internet. Every aspect of our environment "must" be intelligent, graphical and seamless. This is the dream that software companies like Microsoft peddle -- and we're the ones who buy it.

It's easy to jump on the "pick-on-Microsoft" bandwagon, but the virus security issue runs much

deeper than that. Cohen insists that it's not about Microsoft, although it may seem that way. "Basic virology teaches us that you need a certain concentration of susceptible population in order for a disease to become pandemic. Microsoft has that, while ... others do not."

Tumpic adds, however, that Microsoft has gone too far in its zeal to create proprietary standards. This, he says, "fuels the madness" and allows viruses to flourish behind the closed doors of proprietary information standards. Cohen might well agree with this sentiment. Like the evil genius behind Jurassic Park, Microsoft has tried to contain life, but life, in the form of computer viruses, is breaking free and asserting itself.

Nurtured behind the closed doors of Microsoft's proprietary data standards, a new generation of viruses is just waiting to burst forth and prey on consumers who are itching to try out shareware, warez ("cracked" copies of commercial software), MP3 files, and active Web content of all kinds. Many users are fighting back, though, and as you would probably expect, Cohen is in the lead.

The solution Cohen advocates is not a "package" that you buy from a vendor and run in the background. It's not convenient, but he has proven by example that it works. It's an aggressive regimen of education at all levels: system administrators, current users, and future users. With education, we can implement a "Combined Defense," where information sharing and change permissions are tightly limited with cryptography, and where a good backup and disaster recovery plan is in place to minimize damage from any possible virus catastrophe. Cohen's defense deals with the real world; catastrophe may be inevitable, but it need not be crippling.

As virus catastrophes go, in fact, the most common strains of the LoveLetter virus probably have more in common with the common cold than with Ebola Zaire. They're annoying, but you'll probably recover. But what about the next e-mail virus scare, or the one after that? Viruses like that may not be sophisticated, but they thrive on user ignorance and if companies like Microsoft are right about what typical users want from their software, there is more and more of that ignorance going around.

Cohen's "tough love" bottom line advice for users who want a secure environment: "Don't use computers without being a real expert." And if you send him a copy of the LoveLetter virus, don't be surprised to get an e-mail back asking you "to send [him] a fax or include the attachment in the body of the message instead." For Cohen, abstinence is a small price to pay for a computer that boots happily every time. Tumpic seems to agree that it's worth it never to have to wake up with Ebola again.

## Relevant Links

[Cure of Computer Viruses](#)

*Fred Cohen*

[Recent Viruses](#)

*Fred Cohen*

[Virus Glossary](#)

*McAfee*

[Amiga SCA Virus info](#)

*O. Meng & A. M. Rojas*

[That's Not a Virus](#)

*Chengi Jimmy Kuo*

[Privacy Statement](#)

Copyright 2006, SecurityFocus