No Clear Answers on Monoculture Issues

James A. Whittaker
Florida Institute of Technology

The Computer & Communications Industry Association (CCIA) recently released a report that questioned monoculture's effect on cybersecurity. Unfortunately, the report clouded this important topic with unsubstantiated claims and language that suggested more of an attack against Microsoft than a warning about the adoption of a universal computing monoculture.

Despite these flaws, the report's premise is interesting and deserves real scientific investigation. The central idea is that if everyone is running the same hardware and software (a monoculture), a single attack could compromise every machine on the Internet. We obviously need to avoid such a situation, but implying that monoculture is the main cause and destroying monoculture the cure is naïve.

The monoculture stack

A computing monoculture is not a new or unforeseen phenomenon: we have been headed toward such a system for many decades. As the computing infrastructure grew from government and academic labs to business and, finally, to the general population, researchers took great pains to make systems interoperable. This was hardly a plot by any specific software company. We have purposefully sought seamless communication between divergent systems to make computers easier to work with. This has created several monocultures at various layers of the Internet. Consider at least three:

- The *networking-layer monoculture* consists mainly of Cisco products—routers, switches, and other networking appliances—running Cisco's Internetworking Operating System (IOS). A monoculture here creates the potential for attacks against the Internet's very backbone.
- The *communication-layer monoculture* arises from the existence of standards that every manufacturer agrees to implement. Programming to a common standard often results in similar implementations, which can then be vulnerable to a single threat. The recent vulnerability identified in the simple network management protocol (SNMP), CERT advisory CA-2002-03, is one example in which the reference implementation itself was at fault, making the number of susceptible machines potentially enormous.
- The *application-layer monoculture*, the only one the CCIA report addressed, consists of Microsoft Windows desktops. There can be no doubt that Microsoft has tremendous market share here. However, monoculture is more complicated than one company's products dominating the market.

Monocultures can exist because of shared specifications, shared internal algorithms, identical exposed interfaces, a common code base, and so forth. This complicated issue can span multiple vendors and products; it does not necessarily equate to a monopoly.

Faulty metaphor

The term monoculture implies widespread susceptibility, but the Internet does not require widespread infection for disaster to ensue. Consider, for example, that many of the "important" machines on the Internet run some variety of Unix. An attack that brought down only the 13 root servers that manage worldwide DNS tables would have devastating consequences, despite the incredibly small scale.

The authors of the CCIA report used a fundamentally flawed analogy: in a biological monoculture, a single disease could wipe out enough of the population to put an entire species at risk of extinction, but computers don't work that way. An attack against only a few machines, like the root servers, could have much more damaging consequences than wiping out large numbers of Windows desktops.

The biological analogy simply does not work well. Consider the SQL.Slammer worm that infected only 75,000 of the Internet's 175 million machines. That's less than one-tenth of 1 percent, but the worm's negative consequences were enormous. Once again, it isn't the number but rather the type of machines infected, and the services they provide, that define the damage inflicted.

Diversity's real impact on the Internet has yet to be fully examined, and we must deal with the tension that exists between the desire for security and the need for interoperability. Given that we do not yet know how to cost-effectively field completely secure systems, we need to better understand what diversity offers as a solution. How much diversity is enough? What if half the world's machines were Macs? Would the Mac then become a target of more attacks? Would its perceived reliability decrease because it became a target?

Do we necessarily have to achieve diversity by litigating companies like Cisco and Microsoft? There is no reason that we cannot insert diversity into otherwise identical systems. Not every Windows machine needs to be a Web server, database server, and so forth. Nor does every Windows machine need identically installed components.

The world is much more complicated than the CCIA report authors would have us believe. Why implicate monoculture and ignore other glaring issues? Consider the following contributing factors to cyber insecurity, none of which is vendor specific:

- *The computing industry relies on a flawed programming language.* Windows, Solaris, Linux, and the rest are all written in a single, fundamentally flawed, programming paradigm. The design of C (and its derivative, C++) never took security into account, and the runtime libraries that support it are well known for their built-in vulnerabilities. The operating systems and applications built on this code all inherit these vulnerabilities and require special attention by developers to program around them. An arguably more secure programming language called Ada was developed as a C replacement in the 1980s, but the entire industry ignored it. For many years, the US government required its contractors to use Ada, but they succumbed to the industry preference for C in the end. Even as you read this article, programmers are likely rewriting secure Ada applications in C.
- *The computing industry relies on flawed development practices.* Development practices have come a long way in the past few years thanks to various process-improvement efforts, but the code-and-fix paradigm that reigned for more than three decades cannot be undone overnight. Many vendors still use this approach to produce software quickly and then issue patches to correct problems. When security and robustness are not issues, it is easy to justify this paradigm: users' need for functionality and vendors' need for profit can be the focus when quality is not a major concern. Indeed, this attitude has prevailed in the majority of software companies.
- *Novice users are entering the computing community in unprecedented numbers.* Security is not just a technological problem. Preventing novice users from doing stupid things is a difficult endeavor. We can scare these users away by withholding usability features, or we can embrace them by providing good training and making it harder to perform insecure activities. But either way, it is difficult to absorb masses of untrained and uneducated (technically speaking) users without their naïve behaviors exacerbating security incidents.

The facts concerning the effect of monoculture are ambiguous, at best. Until the scientific community runs real, objective experiments to test these conjectures, we'll never know the truth. Perhaps the CCIA report can be used as a wake-up call—not for the artificial break-up of a monoculture, but for a real discussion of the underlying issues that affect the stability and security of the computing infrastructure. □

James A. Whittaker *is a professor of computer science at the Florida Institute of Technology. His research interests include security testing, malicious code, and anti-cyberwarfare. He received a PhD in computer science from the University of Tennessee and is the author of* How to Break Software *(Pearson Addison Wesley, 2002) and coauthor, with Herbert H. Thompson, of* How to Break Software Security *(Pearson Addison Wesley, 2003). He is also an* IEEE Security & Privacy *editorial board member. Contact him at jw@cs.fit.edu.*