01 Jan 1989

Safe Computing

Bruce M. McMillin
Missouri University of Science and Technology, ff@mst.edu

T. L. Casavant

Recommended Citation

B. M. McMillin and T. L. Casavant, "Safe Computing," *IEEE Potentials*, Institute of Electrical and Electronics Engineers (IEEE), Jan 1989.
The definitive version is available at https://doi.org/10.1109/45.41535

Safe computing

Thomas L. Casavant
Bruce M. McMillin

How "diseases" are transmitted, and safety measures network partners can take

Back in graduate school we had an office next to a large field. Around fall the mice would get cold and come into our office for warmth. This invasion was compounded by an enormous array of snacks an office mate kept in his desk. The snacks made the office particularly attractive for the little creatures. The mice didn't really bother us since they would scoot through the office late at night. Each morning, our office mate, however, found more snacks gone and little brown presents left by the mice in his desk. He took it upon himself to safeguard his goodies. First he attempted to plug all the holes in the office by stuffing every nook and cranny with aluminum foil. Anyone who's ever had mice in their house realizes the fallacy in this move; the mice just find new holes. When this process proved ineffective, he began to set mouse traps. This realized impressive results; we caught several mice per day for weeks. The mouse count even made it into the departmental system's log-in message. After a while the number of catches fell off—but not the mice droppings in his desk.

The purpose of this article is not to describe rodent problems nor is it to preach on the evils of snack food. Rather, we are interested in the problems of unwanted benign or malicious "visitors" in a computing system connected to a computer internetwork. The popular press has been inundated with versions of "virus attacks" and "worm attacks." Unfortunately, the analogies of these attacks drawn by the popular press, as well as by some computing professionals, to their biological counterparts has unnecessarily increased the mystique surrounding these attacks. We grudgingly acknowledge these labels of worm, virus, and trojan horse for the purposes of discussion and give technical definitions for each in Figure 1.

The now infamous "Christmas tree worm" of 1987 led off the most recent wave. A user composed a program that, when executed, displayed a view of a Christmas tree and a greeting. This program was sent to several users on the BITNET computer network, who also ran this program. As a side effect, however, the program looked at all the electronic mail distribution lists of the user and forwarded the message to each address, much as the geometric progression of a chain letter. The internetwork was swamped with greeting messages and, thus, effectively shut down attached networks for several days. A more recent attack occurred on the DARPA Internet* in November 1988. This, too, was a worm attack that exploited a well-known "hole" in the electronic mail facility of computers running the UNIX operating system. The worm was able to gain access to many UNIX systems and run many programs thus slowing down the systems for all its normal users.

Why is it so easy to create these worms? How did they come to such prominence? Are they bad? If so, how can we provide safe computing in the sense that unauthorized or

*Technically the DARPA Internet interconnects major government labs such as NSF, DOE, and NASA and most research institutions. The popular name ARPANET comes from the network created by the Advanced Research Project Agency (ARPA) in the 1970's. The name of the organization later was changed to the Defense Advanced Research Projects Agency (DARPA). DARPA now oversees the DARPA Internet.

Is it a worm, virus, or trojan horse?

worm

A worm is a program that propagates itself from computer to computer. Worms may be benign or malicious. A beneficial worm might be a migration of computing workload to unused computers. An example of a malicious worm is the November 1988 DARPA Internet intrusion in which a program rapidly replicated itself into many computers connected by a network.

virus

A virus is a piece of code that can incorporate itself into other programs. It is hidden into a program in the same way as a trojan horse and can propagate in a similar manner to a worm.

trojan horse

A trojan horse is piece of code embedded in a program that performs unwanted actions. A sample program might be a log-in program that not only performs the log-in function but also makes unauthorized user name and password copies.

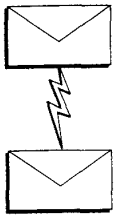
Fig. 1. Definitions.

0278-6648/89/0010-0029$01.00 © 1989 IEEE

Why do we want an internetwork?

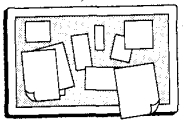
If internetworks have such tremendous security problems, why do we even bother? To answer this challenge, consider some of the applications that require a computer internetwork.

electronic mail



To be able to send electronic mail is to be able to send a message within a few seconds to anywhere in the country and to many parts of the world. It is easier to send someone electronic mail than to try and play "telephone tag," i.e., repeated attempts to contact someone who is always "away from their desk." Electronic mail helps to ease communication between parties without the formalism of a paper letter. It is much easier to send electronic mail than to actually call someone whom you have never met before. The forum of electronic mail allows you to compose your thoughts and allows the recipient to thoughtfully formulate a reply.

bulletin boards



Electronic Bulletin Board Systems (BBS) are computers that allow users to "post" articles/information ranging from technical to non-technical for other users to read. This, for one, enables rapid dispersal of new information and encourages discussion of topics at a nationwide level; users may "follow-up" on a topic, thus creating a discussion among users who never meet face to face.

remote execution

There are several supercomputing centers established throughout the United States attached to an internetwork. Researchers may submit programs to, and collect results from, these supercomputers from their own local workstations without ever leaving their offices. The savings in travel time are enormous. Furthermore, since supercomputers in a center may be shared, this scheme allows for more efficient allocation of computing resources.

Fig. 2. Internetworking applications.

unexpected intrusions are not possible? To answer these questions, we must examine the vehicle that allowed these attacks to occur, namely the open computer internetwork.

An *internetwork* allows for the interconnection of various types of physical networks, each with differing technologies, into a single functional coordinated unit. This means that users can easily perform network applications (see Figure 2) such as electronic mail and electronic bulletin boards. Researchers may submit computations to specialized computers hundreds of miles away and receive the results of their computations at their local workstations. The internetworking software forms an open systems interconnection in which systems of different architectures can cooperate to form a single internetwork.

Members of an internetwork agree to provide certain *services* to other members of the internetwork. Members of the DARPA Internet use a standardized agreement for information exchange called the Internet Protocol or IP. The TCP (Transmission Control Protocol) utilizes the IP to send and receive messages at well known *ports*. A port is an address at which a service resides. For example, the ability to send and receive electronic mail is maintained by a specific port within the TCP. The ability to transfer files is provided at another port. All in all there are several hundred ports reserved for use by well-known services. On the down side, each one of these ports is, in principle, a point at which a system may be entered or compromised.

How does secure communication against maliciousness fit into this scheme? In both of the examples presented, it would have been easy to make the attacks malicious (for example deleting or modifying files) thus causing great damage; nothing in the system security prevented a malicious attack, it was the worm author's decision not to be malicious. The formal study of computer network security, for the most part, has concentrated on preventing attacks on the *association model* (Figure 3) of computer to computer communication. The problem with virus and worm attacks is that they do not follow the association model. The endpoints of the communication are no longer secure. Thus virus and worm attacks can pose a real threat.

Professional responsibility

As scientific and engineering professionals, we have a collective responsibility to police ourselves, much as the medical community does. In the past, violations of computing systems were seen as a "test of expertise," and successful violators were regarded with some awe by their colleagues. This view must change. After the DARPA Internet attack, the governing boards of two major computer networks, BITNET and CSNET, issued a joint statement on issues of computer security. Of primary concern to them were statements made by many computing professionals in response to the incident:

> *. . . (We) have been struck by the fact that many public comments on the event have contained statements such as, "We learned from it," "We*

Guarding flank by the book

The association model assumes that the attack will be made on a well-defined communication path called an *association* between two services at their ports. Attacks may be passive or active. A passive attack involves obtaining information in an unauthorized way by "listening" to an association. Such information might be system passwords, payroll data, social security numbers, and so forth. In an active attack, an intruder might synthesize bogus messages and insert them into the association, play back messages from a previous association at some later time, or deny the receiving member all of its messages.

The primary tool for protection from attack is cryptography in which messages are protected by secret codes known only to members of the association and which are difficult for an intruder to obtain or guess. Encryption of a message protects it from a passive attack. Protection from an active attack can be achieved by requiring each member of the association to identify itself to each other member of the association through a private encrypted digital signature. This ensures that an intruder cannot forge the identity of a valid member of an association. Denial of service may be guarded against by requiring both participants in the association to exchange some predetermined secret information at the start and end of the association. To prevent the playback of an old association, the current date and time may be encrypted as part of the initial exchange to ensure the association is current.

Fig. 3. The association model.

will make sure technically that it will not happen again," or "He did us a favor by showing . . .," unaccompanied by expressions of ethical concern . . .

They go on to make the following comment:

We condemn the perpetration of such "experiments, games, or features" by workers in our field, be they students, faculty, researchers or providers. We are especially worried about widespread tendencies to justify, ignore, or perpetuate such breaches. We must behave as do our fellow scientists who have organized around comparable issues to enforce strong ethical practices in the conduct of experiments.

Clearly the point being made is that computer networks are vital economic systems whose disruption could cause a significant nationwide societal impact. The time when computer security violations were written off as "pranks" has passed.

A debate exists as to whether, when a security problem is found, to keep the flaw a secret or publish it. The former group sees publication of existing security holes as a threat that simply invites "exploration." In some respects, this is a valid viewpoint. If it were found that one could break into a bank vault by simply kicking it in at some obscure place, mentioning this fact to the general public would not be wise. On the other hand, such a situation certainly should be rectified. The latter group, in the interest of removing the problem by engaging the resources of the computing community to find a solution, is perhaps the most widely held view. While, in view of the condemnation by the BITNET/CSNET boards, the first view might be appropriate, in reality, the second view is more plausible. We simply must assume that malicious intruders exist and take actions to protect ourselves.

Protection

When you hook your computer to a network, you are hooking up to every computer that computer has been with . . .
"Saturday Night Live," 1988

How can we protect systems from unauthorized intrusion? The amount of protection provided is in inverse proportion to the ease of usability of the system. Figure 4 illustrates some options ranging from "safe" to "take your chances."

Achieving safe computing is difficult if your computer is connected to an internetwork. The benefits of network interconnection, however, tend to far outweigh the disadvantages. This brings us back to the mice problem, which finally did get resolved. Putting the snacks in glass jars and letting the mice run innocuously around our feet was the ultimate solution (we could hardly put our office in a glass jar). The moral? Protect what you *really want* to keep safe; the rest doesn't really matter.

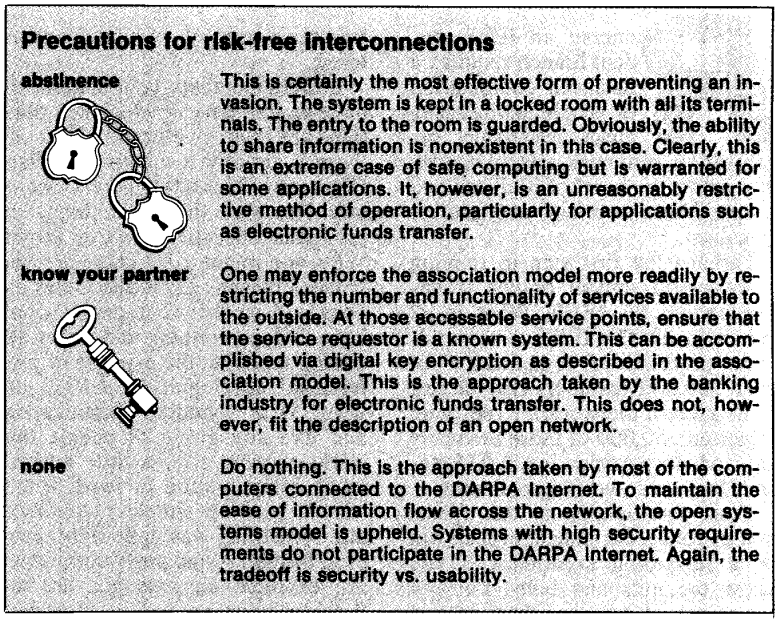


Fig. 4. Some options for "safe computing."

Read more about it

- Voydock and Kent, "Security Mechanisms in High-Level Network Protocols," *ACM Computing Surveys,* 15 (2).
- Morrie Gasser, *Building a Secure Computer System,* Van Nostrand Reinhold, New York, 1988.
- Comer, Douglas, *Internetworking with TCP/IP: Principles, Protocols, and Architecture,* Prentice-Hall, Englewood Cliffs, NJ, 1988.
- Eugene Spafford, *The Internet Worm Program: An Analysis,* Purdue University Technical Report, November 1988.

Acknowledgments

This work was supported in part by the National Science Foundation, in part by the AMOCO Faculty Development Program, in part by the NSF Software Engineering Research Center (SERC), and the Supercomputing Research Center (SRC).

About the authors

Thomas L. Casavant received his B.S. degree in Computer Science from the University of Iowa in 1982, and his M.S. and Ph.D. degrees in Electrical and Computer Engineering from the University of Iowa in 1983 and 1986, respectively. From 1986–1989, he was an Assistant Professor of Electrical Engineering and Director of the Parallel Processing Laboratory at Purdue University. He is currently an Assistant Professor of Electrical and Computer Engineering at the University of Iowa. His research interests include computer architecture, parallel processing, operating systems, distributed systems, and performance modeling and analysis.

Bruce McMillin received his B.S. in Electrical and Computer Engineering and his M.S. in Computer Science from Michigan Technological University, Houghton, Michigan, in 1979 and 1985 respectively and his Ph.D. in Computer Science from Michigan State University, East Lansing, Michigan, in 1988. He is currently an Assistant Professor of Computer Science and Director of the Experimental Computation Laboratory at the University of Missouri at Rolla. His current research interests include fault tolerance, multiprocessing systems, parallel algorithms, software engineering, and distributed systems theory. □