# Causes & Remedies for Social Acceptance of Network Insecurity

Mike Fisk
*mfisk@lanl.gov*
Los Alamos National Laboratory / University of California San Diego

May 10, 2002

**Abstract**

It is widely accepted that computer systems could be designed much more securely that they currently are, so why is it that quality does not seem to be improving? In this paper we will argue that security is being held back not by technical problems, but by social acceptance of network insecurity. We argue that only legal and economic systems such as liability and insurance have the potential to change the status quo and improve the state of network security.

## 1 Introduction

Many researchers and even more practitioners have built careers on detecting and responding to network attacks and vulnerabilities. But all of this effort addresses symptoms of a larger problem that goes largely unaddressed. The vast majority of attacks make use of software vulnerabilities that are entirely preventable. While the state of formal methods for software validation is still rather inadequate, there are well known technical and procedural techniques for preventing these vulnerabilities. But applying these techniques can be resource intensive and will not be done without sufficient incentive. In this paper we argue that the industry is currently in a sub-optimal, but self-supporting equilibrium that does not support the effort required for software improvements. Further, we argue that solution to this problem is economic and social rather than technical. Specifically, we argue that without legal liability there is insufficient incentive for parties to fix security problems. We examine how the insurance industry could be an important economic force in the remedy of this system, but also introduce novelties of the software industry that present barriers to applying historical insurance system models.

### 1.1 Where should the problem be fixed?

The most visible defenses against attacks from the Internet are devices like firewalls that are deployed in strategic locations in the network where they can protect hundreds or thousands of systems at a time. But how many problems would a thousand systems have? The actual number of exploited problems is quite low. The issue is that the same problems are exploited over and over again on different systems. The most frequently exploited vulnerabilities are software flaws in common software products. The most economical place to correct these flaws is not on each of the millions of installed systems, or as a network protection on thousands of networks, but just once in the product itself.

For operators of individual systems, the cost/benefit relationship does not make individual system security a compelling problem. One of the primary reasons that consumers accept network insecurity is that is usually viewed as an annoyance rather than a substantial cost. Many individuals (both at home or at work) perceive a low level of risk from network attacks. The impact of a break-in is often invisible and the usually pre-existing ability to backup and restore the system often addresses the residual risk.

But for institutions that are responsible for large numbers of systems, it is almost certain that some unprotected systems will be compromised. For institutions worried about information security[1] any compromise can be worrisome. Further there is significant workload associated with restoring systems that have been tampered with and in responding to complaints from other netizens who are victimized by compromised hosts. The result is that even though individual users may perceive the risk as small, the probability of loss over a larger organization is large enough that it cannot be ignored.

Finally, software authors frequently have millions of copies of their products installed around the world. The amount of effort that it takes them to correct or prevent problems is actually very little compared to

[1]Even universities have had theft and misuse of unpublished or proprietary research.

the number of systems that it will affect. Unfortunately, software authors can shift this burden to their customers at no cost of their own. As long as consumers are accepting of this situation, there is no reason for software authors to pay more attention to security.

## 2 Supply & Demand from the Adversary's Perspective

While we typically think about the problem from the perspective of the defender or the victim, it is equally interesting to examine the problem from the perspective of the adversary. In particular, the supply and demand of exploits captures many of the dynamics of the community.

Attackers need a continuous availability of exploits, because several processes act to close exploits over time. First, many exploits are published and corrected in future software releases. Second, the over-use of exploits makes them high-profile and often results in the creation of firewalls that block or mitigate attacks. Fortunately for attackers, discovering new exploits is not very difficult. Given a large community of people searching for them, the supply of previously undocumented attacks seems practically endless. Each of these attacks usually corresponds to a flaw in the design or implementation of a software system.

The exploit pool operates as a system with supply and demand effects. A large class of attack consumers known as 'script kiddies' are eager to utilize attacks, but generally lack the intellectual capability to create their own. However, there is no shortage of able-minded people searching for new exploits. The primary incentive for people to independently search for and publish exploits is to promote their own skills to a community. Everybody from consulting companies to school kids use this as a metric within their communities. And this community succeeds in finding new exploits regularly.

Effective prosecution of attackers could deter potential attackers and reduce demand as a result. Unfortunately, it is very difficult to identify, establish attribution, and prosecute these individuals. This complexity is partially technical and partially political. Attackers often work through several layers of indirection through different institutions or countries. Locating the human behind the attack requires tracing back through this path. By involving multiple institutions and countries, an investigator must manually negotiate with each hop along the way to gain information about the previous hop. Prosecution across national boundaries is further complicated by the lack of widespread extradition treaties. As a result, prosecution has failed, to date, to act as a significant deterrent.

## 3 Supply & Demand from the Defender's Perspective

In this section we reason about the supply and demand of secure software. This relationship is of primary concern to the defenders and potential victims of Internet exploits. Software quality is a research area in its own right. Given the current state of formal software testing and proving, improving or maintaining the quality of a software system is primarily a manual process affected by the quality and quantity of expertise applied to it. But given enough demand from consumers, software providers could supply improved quality.

Unfortunately, the system is in a state of equilibrium in which customers tolerate poor quality and will continue to use new software with poor quality. The software industry can be described as a *supergame* in which each turn consists of software authors choosing how much effort to put into security and consumers choosing whether or not to use the author's product. Unfortunately, the current state of the industry is a sub-optimal equilibrium in which consumers have learned to tolerate faulty software and vendors can therefore be successful without improving software security.

It is not that software authors are incapable of producing software systems with fewer vulnerabilities, it is just that they are not sufficiently motivated to do so. In many ways these problems can be compared to automobile quality and safety of several decades ago. Lacking influences such as foreign auto manufacturers or government safety mandates, the industry had no incentive to improve their products and customers had no choice but to continue purchasing these products.

### 3.1 Lack of Demand

The disconnect in the state of Internet security is that the operators of the vulnerable systems have little incentive to protect their systems, while victims such as a high-profile e-commerce site have great interest, but no control in protecting the bulk of vulnerable systems.

Computer security is frequently decomposed into three goals, availability, confidentiality, and integrity, with all three applying to information stored on computers and availability also applying to the use of a system as a whole (for instance the ability to make computations regardless of the availability of stored information). This same taxonomy describes the different losses that may occur: loss of use, loss of confidentiality, or loss of integrity.

However, a large fraction of computer break-ins are not directly associated with one of these classic categories. Many attackers are parasitic and do not disturb the availability of the systems they break into or the integrity of the information. Rather, attackers install additional software and use those systems to launch further attacks on other systems. In some cases this is a game of territory-building while in other cases numerous exploited systems may be used simultaneously to create a denial of service attack on a yet another target. The victims of these secondary attacks range from high-profile e-commerce sites to the home computer systems of acquaintances from cyber chat communities.

## 4 Liability: Generating Demand

We argued in Section 2 that the attackers are unlikely to be included in this equation. That leaves us with the problem of creating a sustainable system purely from the remaining parties: software authors, potential victims, and the operators of network that may be used to launch attacks. This section examines some alternative approaches to this problem and argues that operator liability is a more attractive solution than author liability.

The sad state of network security could arguably be improved in the long run if there was legal liability for either building or operating systems vulnerable to exploitation. There is currently a glaring lack of incentive to secure systems. Legal liability could create this incentive and the insurance industry could be expected to create quality improvement forces similar to Underwriters Laboratory and the Insurance Institute for Highway Safety.

### 4.1 Software Author Liability

Since software authors have the biggest opportunity to prevent problems, it seems appropriate to focus on making them responsible for the security of their products. However, there are some unique aspects of computer software that make it challenging to apply traditional notions of product liability.

Software is often sold as a product and the 'manufacturers' could be held liable for product security and safety. However, the software industry seeks immunity through End User License Agreements that disclaim all warranties and through legislation such as the Uniform Computer Information Transactions Act. Further, software is not always manufactured or even sold in the traditional sense. Software is often published or released as the result of paid or unpaid professional labor rather than as a product. Thus the creation of insecure software could be viewed as professional malpractice.

However, there are also challenges to applying standards of malpractice to software. In particular, it is important to understand the important and growing role of *open source software*. Authors of open source software can be viewed as volunteers who donate services to society. Consider a contributor to an open source software project. This contributor receives no income from the contribution but contributes because they or their employer will receive the benefit of being able to use the software for free. As the number of contributors to these projects grows, so does the value of the software. However, the software is often provided for free to anybody who wishes to use it, even if they have made no contributions. Thus, even a high school student can create software in his or her spare time that ends up being used by thousands of companies world-wide. That student receives no income that could be used to offset liability claims. Further, the student would be dissuaded from contributing if they had to purchase malpractice insurance to contribute safely.

Now assume that this student's software has a vulnerability that is widely exploited by an attacker who uses the exploited systems to launch a 4-hour denial of service attack on say, Amazon.com. As a result, Amazon sees a loss of 4-hours of revenue plus additional costs for investigation, recovery, and damage to public image. Then is the student programmer liable for creating insecure software? The software project to which she contributed may not be a legal entity. Is the leader of the project or the project board responsible? They too may receive no financial compensation to offset liability claims.

By way of analogy, should a first-aid volunteer be held to the same standards of malpractice as a paid physician? In 1997, the 'Volunteer Protection Act' became federal law.[2] In short, this act protects volunteers from liability 'for harm caused by an act or omission of the volunteer' unless the plaintiff can prove high standards such as willful or criminal misconduct, gross negligence, or reckless misconduct.

Clearly there are reasons why society might want to protect volunteer programmers as well. Between this complication and the success to date of product agreements that exclude liability, it seems that software author liability is not likely to be the solution.

### 4.2 System Owner Liability

Given these problems with author liability, we instead shift our attention to the system owners. Should someone who places a system on the Internet be liable for damage caused by unauthorized users of that system? Do they have a responsibility to prevent unauthorized use? It is important to remember that the exploit may have not even been public. In this case, the operator may have had no reasonable way to prevent it. Given that the operators of these systems have no way to fully secure software written by third parties, one cannot expect that that unauthorized use will be completely eliminated.

While being held liable for things not fully under our control seems onerous, consider the automobile insurance industry in which vehicle operators are often legally required to carry insurance against events that society describes simply as 'accidents' (even if the operator is found at fault). Accidents are expected and rarely criminal, but injury or property damage are common and society finds it desirable to not penalize the victims of these accidents. So instead, the cost is amortized across the community in the form of insurance premiums. However, the insurance industry acts as a positive force to reduce risk. For example, insurance companies penalize poor drivers and drivers of unsafe cars. They push for new safety features such as airbags or daytime running lamps. Institutions such as the Underwriters Laboratory and the Insurance Institute for Highway Safety become actors in this system. They test commercial products for relative safety and create incentives for products to do well in those tests.

However, what are the real costs of network attacks? Physical damage is rare. The largest losses that are usually quoted are based on loss of revenue due to being inaccessible to customers. Cleanup and recovery costs are more directly measured. Left unbounded, it seems likely that damage claims could be huge. Would the positive influence on product security be proportional to the total losses covered by the insurance industry? If so, then large damage claims may be useful. If not, then perhaps there should be artificial caps placed on the size of damage claims.

Finally, there is a sanity test that an individual with a home computer should be responsible for less of the burden than an institution with thousands of systems that are no more secure than the home computer. Unlike software which can can be replicated infinitely without cost, computers and bandwidth are both commodities. Thus, an insurance policy based on the likelihood of liability would be proportional to the number of systems or the amount of bandwidth available for denial of service attacks. Thus liability and insurance costs would not scale disproportionally to other costs of being on the Internet.

## 5 Conclusion

While the perpetrators of Internet attacks should obviously be held liable for their actions, we have argued that overall security can be more pragmatically improved by also making system operators liable for the misuse of their systems. We accept that operators cannot fully prevent misuse and we therefore advocate an insurance model where liability for an event is expected and accepted without stigma. While it may not be feasible to always hold software authors liable for faults, it is expected that operator liability will pressure authors to improve software quality and may lead to routine, independent audits of software security. The desired result is the creation of a competitive environment in which customers are uniformly interested in security and which therefore provides incentives to both authors and users for security to be improved.

[2]Publicl Law 105-19