In ASLIB Proceedings, 52:3, March 2000, pp. 93-97 (Preprint)

Published in
ASLIB Proceedings
Volume 52, Number 3,
March 2000, pages 93-97

Why isn't the Internet Secure Yet?

J. D. Tygar, University of California, Berkeley

Alma Whitten, Carnegie Mellon University

Futurists promise us that the Internet will soon be the premiere vehicle for communication, information management, commerce, entertainment, and education. Citizens of the world will be united in a single, easy-to-use, universal network that will deliver a new renaissance of democracy and free flow of knowledge, fulfilling the promise of a global village. While these predictions reflect a certain degree of technological optimism, even the most jaded observer will concede that the Internet has fundamentally changed – and will continue to change – the way we access information. And the full flowering of this new golden age will occur just as soon as we make the Internet safe and secure.

But researchers have been working on network security for at least 30 years – and they have delivered powerful technologies: new technologies for cryptography protecting information that is transmitted or stored in a distributed system; new technologies for access control that allow information owners to exactly specify who can access that information; new technologies for electronic commerce that allow intellectual property owners to charge for access to information; and new technologies for protected execution environments that allow remote code to run safely on a host system. And while fundamental research problems still exist, we have technology that is perfectly adequate for securing most contemporary applications.

So why isn't the Internet secure yet?

We argue that a fundamental problem is the *usability of security*. Even when applications have powerful mechanisms built in, they are so difficult to use that most users do not properly configure their security systems. Vendors attempt to make security easy to use by causing pop-up boxes to show up in dangerous or anomalous situations – but the response of most users is to take any action necessary to make the annoying pop-up box disappear. Indeed, one clever programmer has even written software to automatically click "OK" on all pop-up boxes so users of the software will be untroubled by system notices.

But security is not simply a function that can be automated away – it depends heavily on the context and content of the information that is to be protected. While human being can understand the difference in the secure needs associated with files containing medical records as opposed to files containing athletic records, it is not clear that machines will understand this. Moreover, as new threats emerge, machine protections will always be out of date. Worse of all, by automating security, one provides attackers with an algorithm for understanding exactly what steps will be taken in case a system comes

under attack. So we need to find a way to make users be able to be responsible for their own security..

Perhaps the greatest single weapon in the arsenal of security tools is *cryptography*. As is well reported in the literature, cryptography has the power to provide secure communications, protect transactions, provide powerful privacy, and validate the integrity of information. Cryptography forms a formidable tool, but one that will lie fallow if people do not know how to use it. (We have already heard numerous reports that most users are not able to understand – or do not regularly inspect – the underlying cryptographic certificate mechanism offered to show the authenticity of web pages and applications. Reader, do you regularly inspect them?)

Understanding security is made more difficult by the complex melange of different mechanisms used by a variety of applications and web sites. Consider the problem of integrity of software. Perhaps you are careful to only install applications originating from trusted sources. You may be surprised to learn that your system is still vulnerable to attack. Powerful applications such as word processors, spread sheets, and presentation software not only display content but run active programs. These active programs introduce a set of potential risks – it is all too easy to include a computer virus in a Microsoft Word file, for example. If the problem is serious, patches and fixes are inevitably provided by vendors, but require diligence to install them on a regular basis. And these are just stopgap solutions – to safely run executables loaded over the network, you need to be able to supervise and control the behavior of programs on your system. One solution is for you to never view any content that you did not create, but to take such a stand is to abandon the vision outlined at the start of this essay

Or, consider the problem of privacy. Most readers of this essay have probably ordered a book from an online bookseller such as Amazon.com. Were your purchases of those books protected? How can you find out?[1] The answers will be different for each web merchant you transact business with. How can you tell what a site is really doing as opposed to its public stance? In most cases you can not. (These problems have become quite pervasive – the following story is a powerful anecdote of the threats inherent in such loss of privacy: a woman filed a personal injury lawsuit against a major California supermarket chain after she slipped in a puddle of water and had to be hospitalized with a broken hip. The supermarket is alleged to have threatened to disclose in court the customer's purchases, as recorded from her supermarket frequent buyer card, with the intent of showing that she had purchased a large amount of alcohol and was thus likely to have been drunk at the time of the accident.) The solution to these privacy problems is particularly challenging since it requires a combination of clever technology together with a clear, coherent policy stance.

[1] As of October 1999, the answer for Amazon.com can be found after some searching of their web site. The company reserves the right in the future to freely distribute purchase information to third parties unless you send blank e-mail messages to never@amazon.com from each account you use to shop with Amazon.com.

The Internet continues to grow at a frightening rate. At this stage of the Internet's development, most new users to the Internet are not very technologically sophisticated. Thus, the average level of awareness of security and security mechanisms is very likely decreasing unless we take active steps to make users more security savvy across the Internet.

Of course, problems of hard-to-understand software and poor user interfaces are ubiquitous across the field of computing. So why are we putting so much attention on the usability of security and privacy mechanisms? Because the stakes are so much higher with security:

- Security software requires that users understand all mechanisms – even a single weak link in the chain can lead to compromise of information. This is quite different from most software, which does not require a user to understand all mechanisms. For example, perhaps the reader has never needed to explore the intricacies of mathematical typesetting or the use of Japanese characters in his or her favorite word processor. You won't need to understand these until you have to use them. But if you don't completely understand how to generate, protect, and use your cryptographic keys, your information will be subject to attack. For firms, a single employee who doesn't understand the mechanisms can be a risk to the firm's entire site.

- Once disclosed, private information can not be recovered and made private again. One can protect a site after it has been attacked, but this is rather like closing the barn door after the cows have left. This is different from most software which is more forgiving – e.g., a mistake made in editing a document on a word processor can always be fixed at a later date.

- Users may not even realize that something is amiss with the security of their system until a disaster strikes. For example, if your communications are subject to eavesdropping, you probably won't realize it until those communications are monitored and then disclosed. Even more frightening is the prospect that communications are monitored and not disclosed. This is different from most applications, which more immediately reveal when a problem exists. To again use the example of a word processor, a document typesetting error will usually be quite obvious to the user.

How bad are things, really?

Nobody claims that computer security is easy for people to understand and manage, but how terrible are these problems in real, day-to-day life? When an ordinary person tries to protect her privacy and security while using the Internet, does she face a task that is tediously confusing and inconvenient, but one that can be managed if necessary? Or is it the sad truth that most security software is simply impossible for most people to manage at all?

To get some answers to these questions, we assembled a group of twelve people who were experienced users of computers and particularly of e-mail. All twelve had at least some college, and several had advanced degrees. Some were artists, some worked in medicine, some were administrators, and some did computer programming. Our reasoning was that if this group of diverse yet relatively competent and accomplished people couldn't successfully make their e-mail secure and private, then the outlook was bleak for the general population to be able to do so.

We gave these people the software that had the most elegant graphical user interface for e-mail security and privacy: PGP 5.0 combined with the well-known e-mail program Eudora. We gave them a printed user manual for them to consult as necessary. Then we gave them a secret e-mail message and a list of e-mail addresses, and asked them to send the secret message, digitally signed and encrypted, to the addresses on their list.

The results were enlightening, to say the least[2].

Three of them promptly sent the secret message off without any encryption to protect it, mistakenly believing either that they had encrypted, or that the software would magically and invisibly apply the encryption on their behalf. One did not even realize his error after the fact.

One person never managed to figure out how to encrypt at all, even after an hour and a half of trying. Seven of the others became confused about the basics of how public key cryptography works, and used the wrong keys to encrypt, so that the people they sent the secret message to could not decrypt it. Only half of the group managed to eventually send any correctly signed and encrypted e-mail message, and only four of them succeeded in sending the original secret message, digitally signed and encrypted, to the addresses on the list as we had asked. Many problems, pitfalls, and points of confusion had to be overcome en route to this limited success, often leading to the expression of great frustration.

We wanted to know how bad the problem was: whether generally competent people could manage e-mail security if they were willing to put in reasonable effort, or if even the best-designed existing software was unrealistically hard to understand and use. The answer seems clear: the problem is bad. Security is so unintuitive and confusing to most people that we need a new vision of how to make it usable: something beyond the kind of user interfaces that work acceptably for word processors and spreadsheets. Something very simple and clear, something straightforward and definite in its depiction of how each security mechanism works and how to make use of it.

What the future holds

[2] For those who are curious, a formal report of this research is available in *Why Johnny Can't Encrypt: A Usability Evaluation of PGP 5.0*, Proceedings of the 8th USENIX Security Symposium, August 1999. Additional information is also currently available at http://www.cs.cmu.edu/~alma.

As problems in network security continue to proliferate, we believe that pressure will grow for better security user interfaces. But, given the challenges outlined above, are better user interfaces really possible? We believe so. Here is the set of design strategies we think are most promising:

- **Simplify, simplify, simplify**

  Security software is not immune to the market-driven process of feature creep, which causes programs to increase in size and complexity with each successive release, as the latest variations and buzzwords are incorporated into the functionality of the program. Nor is it exempt from the desirability of catering to the needs of a wide range of users, from the simple needs of novices to the complicated and advanced requirements of experts. Size and complexity, however, work against the crucial goal of designing security that the maximum number of users will be able to understand quickly, clearly and effectively.

  How can security software be made simple to understand while still allowing for the needs of a wide variety of users (and of marketing departments)? By carefully identifying just what practical use beginners will actually need to make of the security functions, we can construct a basic and coherent conceptual model of the security mechanisms that they will need. All other functionality and data can be pushed into one or more levels of background, so that the advanced features are available to experts but do not confuse or overwhelm beginners. Structuring the functionality of the security into multiple levels that correspond to clearly identified conceptual models also allows us to make the best use of our other design strategies, as we describe below.

- **Communicate clearly**

  Once we have the security functionality structured into multiple levels, we are well placed to focus on how best to quickly and effectively communicate the workings of the novice level to users. The security mechanisms in the novice level must be presented in terms of a unifying and accessible conceptual model, which should be reinforced as each aspect of the security system is displayed.

  Visual metaphors are of obvious use in presenting such a model, but need to be carefully constructed and tailored so as not to mislead users. In PGP, it is essential to know how certain crucial information used to configure cryptography (the cryptographic keys) are used. And, of course, it is essential to understand the operations done on data to protect it. But PGP doesn't help very much with understanding the keys or the operations. The icons used to represent the cryptographic keys and the encryption operation did not help the users understand public key encryption well enough to know which keys to use, and the quill pen icon used to represent digital signatures and the signing operation did nothing to tell the users that keys are used for signing. Users would have been much better served if the

designers had focused on conveying the basic conceptual model of public key cryptography and tailored their metaphors to reference and reinforce that model.

After designing an initial user interface that optimizes the communication of the necessary basic model, then we can attempt to design the remaining levels to gracefully expand the model as the user comes to need more complex security functionality.

- **Guide and protect**

  A concept from education and, recently, from research in educational software, is *scaffolding*: the technique of providing pervasive and explicit help in the early stages of a learning process, which is then gradually withdrawn (a process referred to as *fading*) as the learner acquires skill and understanding. Our structuring of security functionality into multiple levels which are only gradually accessed by users is itself a form of scaffolding, in which the beginner is helped by being given only a limited and simplified model, to protect against the risk of confusion and error. Scaffolding of this type is sometimes called *training wheels*.

  Other types of scaffolding are also necessary and appropriate for security user interfaces, and the multi-level structuring we propose provides a natural framework on which to build and fade additional guidance and help. Warning messages and other kinds of brief tutoring (in the form of interruptions) that can be presented aggressively to novices and faded as users become more expert. Users can be allowed to turn off such messages as they become annoying, providing another natural fading mechanism, if the messages first provide educational content and are not merely risk flags. This is last point is central and is sadly lacking from much contemporary software.

Will the Internet be secure?

The march towards moving mass information systems to computer networks seems unstoppable. Every day, we read in the paper (or even better, on our favorite electronic news site) about further applications and information repositories available on the World Wide Web. But in ten years, will the Internet be significantly more secure than it is today? We hesitate to answer, since speculating on the future development of the Internet at times resembles astrological forecasting. But of one thing we are certain, we'll be paying a cost. Either we will pay the cost to build and deploy more effective, usable security systems, or we will pay the cost in the form of lost privacy and security. It is easier for an individual to protect privacy from the beginning than to try to recover it once it has been lost. The same holds true for a society.

Biographies

J. D. Tygar is Professor of Computer Science and Information Management at the University of California, Berkeley. He works in computer security and electronic commerce. He has designed a number of systems, including cryptographic postage

indicia (which are the foundation for the US Postal Service's new PC Postage standard for printing out postage on personal computers) and the NetBill micropayment system (now licensed for use by CyberCash.) He has designed cryptographic protocols and systems for protecting privacy and security. He has written and consulted widely in the field. Prior to taking the position at Berkeley, Professor Tygar was on the faculty at Carnegie Mellon University. He received his A.B. from UC Berkeley and his Ph.D. from Harvard University. You can find more information about him at www.cs.berkeley.edu/~tygar.

Alma Whitten is completing her Ph.D. dissertation at Carnegie Mellon University in the area of security interfaces. She received her undergraduate degree in computer science and engineering from the University of Connecticut, and her background includes work in theater, art, anthropology and psychology. You can find more information about her at www.cs.cmu.edu/~alma.