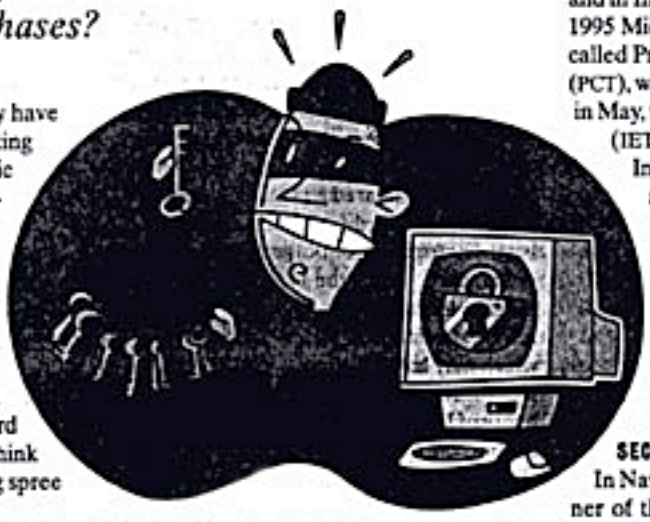
Internet Security Standards

How safe is it to send your credit card number over the Internet to make purchases?

By Michele Rosen



The Internet phenomenon may have happened quickly, but it's taking quite a bit longer for electronic commerce to take hold. Although surveys indicate that almost a quarter of Internet users have made purchases by sending their credit card numbers over the Internet, the same surveys show that many more people still believe that doing such a thing is just plain foolish. Although federal law limits consumer credit card liability to $50, it's disconcerting to think of some hacker going on a shopping spree with your credit card.

Whether or not the fear is irrational, it is certainly understandable. Many also accuse the Internet of being difficult to navigate, but the concepts behind hypertext are child's play compared with the technology needed to protect data from prying eyes. So what really happens when you type in that 16-digit number and press *Submit*? How do Microsoft's and Netscape's browsers stand between unscrupulous hackers and your credit card? The short answer is *encryption*. But of course, as always with technology, it's much more complicated than that.

ENCRYPTION BASICS

You may remember sending notes to your grade-school friends in which you replaced each letter of the alphabet by the number representing its position (A was 1, B was 2, and so on). You didn't know it then, but you were encrypting your message. The key to interpreting the message was to know what the numbers represented. In cryptography jargon, the *key* is the number that, when plugged into an equation or algorithm, allows you to encrypt and decrypt data.

Unless you're a mathematician, however, that's where the simplicity ends. Scrambling data to make it incomprehensible is difficult enough. Doing this in a way that permits returning the data to its original form requires complicated algorithms that employ esoteric mathematical equations.

There are two broad categories of encryption algorithms—*private-key* and *public-key*. In private-key encryption, users accomplish both encryption and decryption with the same key. For data that remains in one location, this is not a problem. A difficulty arises, however, when data needs to be transmitted. If I want to send you a secure message, I use my private key to encrypt it. Then I send it to you. Now you have the encrypted message. But I need a secure way to send you the key.

Cryptographers Whitfield Diffie and Martin Hellman proposed a solution to this problem in 1976 when they invented public-key cryptography. Their solution is deceptively simple: Use two different but related keys to encrypt and decrypt the data. The encryption key is called the *public key*. Since the key used to encrypt the data cannot be used for decryption, there is no risk of its being discovered by unscrupulous individuals. Decrypting the data requires the *private key*—which, thanks to the existence of the public key, can be maintained in a secure location. If I want you to send me an encrypted message, I send you my public key, which you use to encrypt the message. When I receive the encrypted message, I use my private key—which has never left the (relative) security of my computer—to decrypt it. Problem solved!

As always, there is a catch. Public-key cryptography algorithms can take 1,000 times as long as private-key algorithms to encrypt or decrypt data. Public-key cryptography also requires keys up to 10 times as long as those for private-key cryptography to provide an equal level of security.

For these reasons, the security protocols used by both Microsoft's and Netscape's browsers take advantage of the benefits of both public- and private-key cryptographic methods, while avoiding the disadvantages. The most widely used security protocol is called Secure Sockets Layer (SSL). It has been included in Navigator since the first version and in Internet Explorer since Version 3.0. In 1995 Microsoft proposed another protocol, called Private Communications Technology (PCT), which was also included in IE 3.0. And in May, the Internet Engineering Task Force (IETF)—the organization that codifies Internet standards—began considering a new protocol based on SSL, called Transport Layer Security (TLS). Since all three are similar, we'll illustrate how SSL works step by step and then explain how PCT and TLS differ from it, so that you can make a more informed judgment about the security your browser provides.

SECURE SOCKETS LAYER

In Navigator 3.0 and 4.0, the lower-left corner of the screen is reserved for a security icon—a chain in the earlier version and a padlock in the current one. When the broken chain becomes whole or the padlock closes, you know you have entered a secure session with the server you are contacting.

When the browser first connects to a secure Web page, the server sends a "hello request" message. To initiate the secure session, the browser must respond with a message called a "client hello," and the server must answer that with a "server hello." During this initial phase, the browser and server are communicating security information using the handshake protocol, the first part of SSL. The client "hello" message contains a number, called a *session ID*, that uniquely identifies this session between the browser and the server. The message also tells the server which cryptographic algorithms, SSL version, and compression methods the browser supports. Finally, it includes a random number generated by the browser. The server "hello" message responds with the compression method and encryption algorithm it has selected from the choices provided by the browser, the appropriate SSL version, a different random number, and an acceptable session ID number.

At this stage the client and server can exchange digital certificates, which verify that the two parties are who they say they are. The server's certificate can also include a public key appropriate to the public-key encryption algorithm selected during the handshake protocol. This key will be used only for a short time, however; the actual transaction (read: credit card information) will be encrypted