Chapter 1. Introduction to SSH

Many people today have multiple computer accounts. If you're a reasonably savvy user, you might have a personal account with an Internet service provider (ISP), a work account on your employer's local network, and one or more PCs at home. You might also have permission to use other accounts owned by family members or friends.

If you have multiple accounts, it's natural to want to make connections between them. For instance, you might want to copy files between computers over a network, log into one account remotely from another, or transmit commands to a remote computer for execution. Various programs exist for these purposes, such as *ftp* and *rcp* for file transfers, *telnet* and *rlogin* for remote logins, and *rsh* for remote execution of commands.

Unfortunately, many of these network-related programs have a fundamental problem: they lack security. If you transmit a sensitive file via the Internet, an intruder can potentially intercept and read the data. Even worse, if you log onto another computer remotely using a program such as *telnet*, your username and password can be intercepted as they travel over the network. Yikes!

How can these serious problems be prevented? You can use an *encryption program* to scramble your data into a secret code nobody else can read. You can install a *firewall*, a device that shields portions of a computer network from intruders. Or you can use a wide range of other solutions, alone or combined, with varying complexity and cost.

1.1 What Is SSH?

SSH, the Secure Shell, is a popular, powerful, software-based approach to network security.[1] Whenever data is sent by a computer to the network, SSH automatically encrypts it. When the data reaches its intended recipient, SSH automatically decrypts (unscrambles) it. The result is *transparent* encryption: users can work normally, unaware that their communications are safely encrypted on the network. In addition, SSH uses modern, secure encryption algorithms and is effective enough to be found within mission-critical applications at major corporations.

[1] "SSH" is pronounced by spelling it aloud: S-S-H. You might find the name "Secure Shell" a little puzzling, because it is not, in fact, a shell at all. The name was coined from the existing *rsh* utility, a ubiquitous Unix program that also provides remote logins but is very insecure.

SSH has a client/server architecture, as shown in Figure 1-1. An SSH *server* program, typically installed and run by a system administrator, accepts or rejects incoming connections to its host computer. Users then run SSH *client* programs, typically on other computers, to make requests of the SSH server, such as "Please log me in," "Please send me a file," or "Please execute this command." All communications between clients and servers are securely encrypted and protected from modification.

Figure 1.1. SSH architecture