SSH Usage Guide

Matt Gray

1 Introduction

Hello, I've created a quick walk through detailing the features of SSH. Due to the rise of security-related issues in computer networking today, many once-standard protocols have become hopelessly deprecated. I'm talking about telnet and FTP, among others. Passwords are sent over the network in clear text, available for anyone who has the inclination to run a packet sniffer.

SSH is the community's answer to the aforementioned insecure protocols. With SSH, you can do everything you used to do in FTP, and much more.

2 Basic Usage

SSH is more than a replacement for `rsh`, it is a remote-access swiss army knife. In this section, the most common ways of using SSH are introduced.

2.1 Remote Shell

SSH functions extremely well as a remote shell—mimicing `telnet`, with none of the shortcomings. This is also the most common reason for invoking SSH. Here is an example command line:

```
[matt@set ~]$ ssh matt@isis.spa.umn.edu
```

Here, the usage is fairly straightforward. I would like to connect as user `matt` to the host `isis.spa.umn.edu`. Of course, any user and system could be substituted, provided that the user account exists on the destination system, and that the account is authorized to connect.

This simple invocation is all that one needs for opening a remote console on another host. SSH, however, can do much more.

2.2 X11 Forwarding

Sometimes it is desireable to run a remote application that requires a graphical environment (e.g. X windows). Previously, this was possible through the usage of `xhosts` and the `DISPLAY` environment variable. One could log in to a remote machine and redirect display information to a local X server. Unfortunately, this method provides very little in the way of security.

Enter SSH. SSH will compress and encrypt your X11 traffic, and seamlessly allow you to execute remote X applications. The key to this behavior is the following command:

```
[matt@set ~]$ ssh -X matt@isis
```

The `-X` switch instructs SSH to *forward* any X11 traffic over the encrypted connection to your local X server. To the local system, it is as if you ran the graphical program from the command line. This procedure is equally transparent from the remote machine's perspective.

2.3 Summary

SSH's two basic forms allow for remote shell access, as well as the ability to execute remote X applications to display on a local X server. These similar invocations of `ssh` are the meat and potatoes of every day SSH usage—for the average user, these commands are enough to accomplish most common tasks.