Linux supports remote login access through several different servers, including  
Virtual Network Computing (VNC), and even X. Unfortunately most of these methods  
suffer from a major drawback: they transfer data over the network in an *unencrypted* form.  
This fact means that anybody who can monitor network traffic can easily snatch sensitive  
data, often including passwords. (VNC and a few other protocols encrypt passwords but not  
other data.)  
This security limitation keeps these remote login tools from being useful. If using  
a remote access tool means that you’ll be giving away sensitive data or compromising your  
entire computer, it’s not a very useful tool.

SSH was designed to close this potential major security hole by employing strong  
encryption techniques for all parts of the network connection. SSH encrypts the  
password exchange and all subsequent data transfers, making it a much safer protocol  
for remote access.  
In addition to encryption, SSH provides file transfer features and the ability to  
*tunnel* other network protocols. This feature enables nonencrypted protocols to piggyback  
their data over an SSH connection, thus delivering SSH’s encryption advantages to  
other protocols.

The main drawback of SSH is that the encryption and decryption consumes CPU time.

This fact slows down SSH connections compared to  
those of direct connections and can degrade overall system performance.  
If you tunnel a protocol that transfers much more data, you may see a  
greater performance drop when using SSH. However, the improved  
security is generally worth the slight speed cost.

Several SSH servers are available for Linux, but the most popular by far is the OpenSSH  
server (www.openssh.org). This program was one of the first open source implementations  
of the SSH protocol. OpenSSH ships with most Linux distributions.  
OpenSSH may be launched either via the xinetd super daemon or via a startup script.  
Most distributions deliver suitable startup scripts with their SSH packages.  
If you make changes to your SSH configuration, you may need to pass the reload or  
restart option to the startup script, as in /etc/init.d/sshd reload. (Chapter 5 covers  
startup scripts in more detail.) However it’s launched, the OpenSSH server binary name  
is sshd.