CNS /Update Newsletter Feature

Ssh Offers Secure Remote Access to NERSP

CNS Document ID: u980405a
Last Updated: 5/17/99

UF Information Technology

UFIT

2046 NE Waldo Rd, Suite 2100
Gainesville Florida 32609-8942
(352) 392.2061
<editor@cns.ufl.edu>
# Table of Contents

Ssh Offers Secure Remote Access to NERSP ......................................................... 3
Secure Shell Features ......................................................................................... 3
Overview of Secure Shell ................................................................................... 4
Starting Ssh ........................................................................................................ 5
Further Documentation ....................................................................................... 5
Questions ............................................................................................................ 5
Ssh Offers Secure Remote Access to NERSP

This information is for experienced UNIX users. Access to NERSP via ssh (Secure Shell) is now available. Ssh is a program for logging into a remote machine and for executing commands in a remote machine. It is a natural replacement for rlogin and rsh, and provides secure encrypted communications between two untrusted hosts over an insecure network. X11 connections and arbitrary TCP/IP ports can also be forwarded over the secure channel.

Secure Shell Features

Features of ssh include the following:

- **Strong authentication.** Ssh closes several security holes (e.g., IP, routing, and DNS spoofing) and includes new authentication methods: .rhosts together with RSA-based host authentication, and pure RSA authentication.

- **Improved privacy.** All communications are automatically and transparently encrypted. RSA is used for key exchange, and a conventional cipher (normally IDEA, DES, or triple-DES) is used for encrypting the session. Encryption is started before authentication and no passwords or other information are transmitted in the clear. Encryption is also used to protect against spoofed packets.

- **Secure X11 sessions.** The program automatically sets DISPLAY on the server machine and forwards any X11 connections over the secure channel. Fake Xauthority information is automatically generated and forwarded to the remote machine; the local client automatically examines incoming X11 connections and replaces the fake authorization data with the real data (never telling the remote machine the real information).

- **Arbitrary TCP/IP ports can be redirected through the encrypted channel in both directions (e.g., for e-cash transactions).**

- **No retraining is needed for "everyday" users; everything happens automatically, and old .rhosts files will work with strong authentication if the client administrator installs the host key files.**

- **Never trusts the network.** Minimal trust on the remote side of the connection. Minimal trust on domain name servers. Pure RSA authentication never trusts anything but the private key.

- **Client authenticates the server machine via RSA at the beginning of every connection to prevent trojan horses (by routing or DNS spoofing) and man-in-the-middle attacks; and the server RSA authenticates the client machine before accepting .rhosts or /etc/hosts.equiv authentication (to prevent DNS, routing, or IP-spoofing).**

- **Host authentication key distribution can be handled centrally by the administration, automatically when the first connection is made to a machine (the key obtained on the first connection will be recorded and used for authentication in the future), or manually by each user for his/her own use.** The central and per-user host key repositories are both used and complement each other. Host keys can be generated centrally or automatically when the
software is installed. Host authentication keys are typically 1024 bits.

- Any user can create any number of user authentication RSA keys for his/her own use. Each user has a file which lists the RSA public keys for which proof of possession of the corresponding private key is accepted as authentication. User authentication keys are typically 1024 bits.

- The server program has its own server RSA key which is automatically regenerated every hour. This key is never saved in any file. Exchanged session keys are encrypted using both the server key and the server host key. The purpose of the separate server key is to make it impossible to decipher a captured session by breaking into the server machine at a later time. One hour from the connection even the server machine cannot decipher the session key. The key regeneration interval is configurable. The server key is normally 768 bits.

- An authentication agent, running in the user's laptop or local workstation, can be used to hold the user's RSA authentication keys. Ssh automatically forwards the connection to the authentication agent over any connections, and there is no need to store the RSA authentication keys on any machine in the network (except the user's own local machine). The authentication protocols never reveal the keys; they can only be used to verify that the user's agent has a certain key. Eventually the agent could rely on a smart card to perform all authentication computations.

- The software can be installed and used (with restricted functionality) even without root privileges.

- The client is customizable in system-wide and per-user configuration files. Most aspects of the client's operation can be configured. Different options can be specified on a per-host basis.

- Ssh automatically executes conventional rsh (after displaying a warning) if the server machine is not running sshd.

- Optional compression of all data with gzip (including forwarded X11 and TCP/IP port data), which may result in significant speedups on slow connections.

- Complete replacement for rlogin, rsh, and rcp.

**Overview of Secure Shell**

The software consists of a number of programs.

<table>
<thead>
<tr>
<th>Program</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>sshd</td>
<td>This is the server program which runs on the server machine. It listens for connections from client machines, and whenever it receives a connection, it performs authentication and starts serving the client.</td>
</tr>
<tr>
<td>ssh</td>
<td>This is the client program used to log into another machine ssh or to execute commands on the other machine. &quot;slogin&quot; is another name for this program.</td>
</tr>
</tbody>
</table>
scp
"scp" securely copies files from one machine to another.

ssh-keygen
This program is used to create RSA keys (host keys and user authentication keys).

ssh-agent
This is the authentication agent. It can be used to hold ssh-agent RSA keys for authentication.

ssh-add
This one is used to register new keys with the agent.

Starting Ssh

Ssh is the program users normally use. It is started as:

    ssh host

or

    ssh host command

The first form opens a new shell on the remote machine (after authentication). The latter form executes the command on the remote machine.

Further Documentation

Online manuals are available using the man command (for example, man ssh). Information for other platforms can be found on the Secure Shell homepage located at URL http://www.cs.hut.fi/ssh/.

Questions

For questions regarding the use of secure shell (ssh) access to NERSP, please write to consult@lists.ufl.edu [mailto:consult@lists.ufl.edu].

Your Comments are Welcome

We welcome your comments and suggestions on this and all UFIT documentation. Please send your comments to:

    UF Information Technology

    UFIT

    2046 NE Waldo Rd, Suite 2100
    Gainesville Florida  32609-8942
    (352) 392.2061
<editor@cns.ufl.edu>