Encrypted Tunnels using SSH and MindTerm HOWTO

Duane Dunston

duane@duane.yi.org

Revision History

Revision 1.01	2001−06−13	Revised by: PDD
Changed date format (YYYY−MM−DD)		

This document describes how to use SSH and the Java−based program MindTerm to create quick, secure, and reliable VPN−like tunnels over insecure networks.

Table of Contents

1. Introduction

For various reasons this brand new release is codenamed the *release* release.

New code names will appear as per industry standard guidelines to emphasize the state−of−the−art−ness of this document.

This document was written when I read a feedback asking for a template to fill in to make new HOWTOs. This template was initially made by extracting the skeletal structure of the Multi Disk HOWTO which is a rather large HOWTO. It then went through extensive editing.

Stating the background is a simple way to getting started writing the intro.

First of all we need a bit of legalese. Recent development shows it is quite important.

1.1. Copyright Information

This document is copyrighted (c) 2001 Duane Dunston and is distributed under the terms of the Linux Documentation Project (LDP) license, stated below. *It's requested that corrections and/or comments be forwarded to the document maintainer.*

Unless otherwise stated, Linux HOWTO documents are copyrighted by their respective authors. Linux HOWTO documents may be reproduced and distributed in whole or in part, in any medium physical or electronic, as long as this copyright notice is retained on all copies. Commercial redistribution is allowed and encouraged; however, the author would like to be notified of any such distributions.

All translations, derivative works, or aggregate works incorporating any Linux HOWTO documents must be covered under this copyright notice. That is, you may not produce a derivative work from a HOWTO and impose additional restrictions on its distribution. Exceptions to these rules may be granted under certain conditions; please contact the Linux HOWTO coordinator at the address given below.

In short, we wish to promote dissemination of this information through as many channels as possible. However, we do wish to retain copyright on the HOWTO documents, and would like to be notified of any plans to redistribute the HOWTOs.

If you have any questions, please contact <duane@duane.yi.org>

1.2. Disclaimer

No liability for the contents of this documents can be accepted. Use the concepts, examples and other content at your own risk. As this is a new edition of this document, there may be errors and inaccuracies, that may of course be damaging to your system. Proceed with caution, and although this is highly unlikely, the author(s) do not take any responsibility for that.

All copyrights are held by their by their respective owners, unless specifically noted otherwise. Use of a term in this document should not be regarded as affecting the validity of any trademark or service mark.

Naming of particular products or brands should not be seen as endorsements.

You are strongly recommended to take a backup of your system before major installation and backups at regular intervals.

1.3. New Versions

This has undergone many revisions as this began as my final project for SANS GIAC certification.

The latest version number of this document can be gleaned from the main Linux Documentation Project homepage or the authors page.

If you have the capability, it would be nice to make the HOWTO available in a number of formats.

1.4. Credits

In this version I have the pleasure of acknowledging:

Patti Pitz for her editing and help with organizing the paper. Doug Eymand for his technical editing.

1.5. Feedback

Feedback is most certainly welcome for this document. Without your submissions and input, this document wouldn't exist. Please send your additions, comments and criticisms to the following email address : `<duane@duane.yi.org>`.

2. Before we start

2.1. Mindterm and SSH Introduction

Businesses, schools, and home users need more secure network services now more than ever. As online business increases, more people continue to access critical company information over insecure networks. Companies are using the Internet as a primary means to communicate with travelling employees in their country and abroad, sending documents to various field offices around the world, and sending unencrypted email; this communication can contain a wealth of information that any malicious person can potentially intercept and sell or give to a rival company. Good security policies for both users and network administrators can help to minimize the problems associated with a malicious person intercepting or stealing critical information within their organization. This paper will discuss using Secure Shell (SSH) and MindTerm to secure organizational communication across the Internet.

Home users and business travelers are accessing company resources and sending sensitive data over insecure networks. *This opens up a whole new area of security issues for System Administrators (Securing the home office sensible and securely)*, especially since the number of corporate users from home with high−speed access is expected to *"more than double from 24 million in 2000 to 55 million by 2005" (Broadband Access to Increase in Workplace). The increase in the number of airports and hotels offering internet access, especially high−speed access, is increasing and is expected to grow in the future (Broadband Moving On Up)*. This can also leave a door wide open for a malicious person to hijack or view a person's Internet traffic and access their companies. The malicious person may not be interested in the work the employee is doing but just want access to a high−speed server to launch attacks, store files, or other uses. Business people are really at high risk because they don't know who's monitoring their Internet connection in the hotel, airport, or anywhere in their travels. Users of the new high−speed connections are usually not taught proper security protocols and some companies don't have the staff to help the home user and business traveler set up secure communication. Individual users and, surprisingly, some companies have a mentality that *"I don't have anything people want"*. This is very disturbing considering the amount of sensitive information that travels across the Internet from an employee's home or from travelers. What's more disturbing is the availability of free software to perform these kinds of attacks and the software's ease of use. Dsniff (http://www.monkey.org/~dugsong/dsniff/) is a freely available program that has utilities that can allow anyone with a networked computer to highjack a local network and monitor what others are doing and grab passwords and other sensitive data. In his book Secrets and Lies: Digital Security in a Networked World, Bruce Schneier states that Technique Propagation is one of the main threats to network security: *"The Internet is...a perfect medium for propagating successful attack tools. Only the first attacker has to be skilled; everyone else can use his software" (Schneier).*

The purpose of this paper is not how to secure computers but how to set up virtual tunnels to perform secure communication, whether sending documents or sending email. Business travelers should read Jim Purcell, Frank Reid, and Aaron Weissenfluh's articles on travel security. Home users with high−speed access should read Ted Tang's article for information on how to secure your computers with high−speed access. I'd recommend the many resources available on www.sans.org, www.securityfocus.com, or www.securityportal.com for tutorials on how to secure your computers and servers.

The way to ensure that sensitive data is transmitted securely and quickly is to use encrypted methods of data delivery. This can be by way of encrypted email, using secure web−based email services, or establishing encrypted tunnels between two computers. Also, easy to setup and reliable software need to be used in order to allow the inexperienced users the ability to quickly establish secure communication channels. Taten Ylonen 's Secure Shell and MindBright Technology's MindTerm are a quick, easy to use, and reliable solution for securing communication over the Internet.

2.2. MindTerm and SSH

SSH (Secure Shell) is a secure replacement for remote login and file transfer programs like telnet, rsh, and ftp, which transmit data in clear, human−readable text. SSH uses a public−key authentication method to establish an encrypted and secure connection from the user's machine to the remote machine. When the secure connection is established then the username, password, and all other information is sent over this secure connection. You can read more details of how ssh works, the algorithms it uses, and the protocols implemented for it to maintain a high level of security and trust at the ssh website: www.ssh.com. The OpenBSD team has created a free alternative called OpenSSH available at: www.openssh.com. It maintains the high security standards of the OpenBSD team and the IETF specifications for Secure Shell (see the Secure Shell IETF drafts, except it uses free public domain algorithms. SSH is becoming a standard for remote login administration. It has become so popular that there are many ports of ssh to various platforms and there are free clients available to login to an ssh server from many platforms as well. See http://linuxmafia.com/pub/linux/security/ssh−clients for a list of clients and Securityportal.com has an excellent two−part article on ssh and links to ports for different platforms available at http://www.securityportal.com/research/ssh−part1.html. There are programs that also use an ssh utility called Secure Copy (scp) in the background that provide the same functionality of a full ftp client, like WinSCP and the Java SSH/SCP Client, which has a modified scp interface for MindTerm. Please read the licenses carefully to determine if you are legally allowed to download ssh in your country. SSH is free for academic institutions please. Please read the licenses available at the ssh.com website.

MindTerm is an ssh client written entirely in Java by MindBright Technology. One of the key practices of developing security software is proper implementation of the underlying algorithms and protocols it uses. MindBright Technology has implemented the ssh protocol very well in this small application file. It is a self−contained archive that only needs to be unzipped into a directory of your choice and it is ready to be used. It can be used as a standalone program or as a web page applet or both. It is available at: http://www.mindbright.se/download/. MindTerm is an excellent and inexpensive client to secure communication to and from a local and remote location. The MindTerm program located at the download address above is available free for non−commercial and academic use, commercial use is available on a case to case basis. However, the modifications made by the ISNetwork *"is based on the MindTerm 1.21 codebase, which MindBright released under the GPL [General Public License −− see http://www.gnu.org]. Since our version is released under the GPL you can use it commercially for free" (Eckels).* ISNetwork's implementation has all the features of MindBright's MindTerm except it has a nicer scp interface for more user−friendly file transfers. MindTerm does have some drawbacks in that it doesn't support UDP tunneling. In order to secure UDP traffic, a program called Zebedee (http://www.winton.org.uk/zebedee/) will work nicely. Zebedee's server and client program is available for Windows and Linux platforms. It is freely distributed under the GPL License too. You can connect to either Windows or Linux machines using Zebedee. MindTerm will not check to see if your system is secure. It is up to the administrators and users to take care of securing the computer systems. It is easy to implement and it is very effective at maintaining the high level of security implemented in the ssh protocol. This paper will show how easy it is to set up and establish secure communication channels for almost any user and by almost any user. Documents, email, and other data communication can be easily and securely sent to users a few feet away or around the world.

2.3. How MindTerm and SSH work together

SSH and MindTerm will work together to use a technique called port forwarding. Port forwarding is forwarding traffic from one host and a given port to another host and port. In other words, the MindTerm application will open a port on the client's machine (local machine) and any connection to that local port is forwarded to the remote host and its listening port over an encrypted ssh session. Whether or not the connection is accepted depends on the type of request you are sending to the remote host. For example, you wouldn't forward POP requests to a remote host listening on port 21 because port 21 is reserved for ftp requests. Port forwarding is also used to allow connections to a server that is behind a firewall and/or has a private IP address. Essentially this is creating a Virtual Private Network (VPN). A VPN is *"a private data network that makes use of the public telecommunication infrastructure, maintaining privacy through the use of a tunneling protocol and security procedures"* (www.whatis.com). The port−forwarding can only be done with TCP services.

---

3. Software Installation

In order to follow along with this tutorial you will have to install a few packages. This tutorial assumes you have ssh already installed on your server or workstation. If not then you can read the documentation that comes with the ssh or the OpenSSH package for installation instructions for your platform. For the examples that follow, OpenSSH was installed on a RedHat 7.0 server and workstation. OpenSSH was installed on RedHat 6.0− 7.0 and worked the same. The client machine used in the following tutorial is a Windows 2000 machine. Windows 95/98, NT 4.0, NT 5.0, RedHat 6.0−7.0 workstation were all tested as client machines and worked the same. On a side note, the exact same MindTerm jar archive was used on all client systems tested.

- SSH or Openssh
- MindTerm
- FTP Client − Any ftp client should work for this tutorial. Ws−FTP and Leech−ftp are the two most popular for Windows.
- Netscape Communicator − or any other mail client should work.
- Optional: NTOP
- Optional: vlock

4. Server and Client Configurations

4.1. Server Configuration

First, make sure that your server is secure. Though traffic is encrypted as it travels over the Internet, it can be sniffed if someone has root access on the local machine and uses a program like ngrep to sniff traffic on a local machine. For example, in conjunction with the dsniff program mentioned above, the following command could sniff all traffic on the local interface network: **ngrep −d lo**. Securing the server is, however, beyond the scope of this paper.

We'll use the POP (port 110), IMAP (port 143), SMTP (port 25), VNC (Virtual Network Computing) (5901+), and NTOP (default port 3000) services for this example. All traffic will be forwarded to each service's respective port on the remote host running the ssh server. All services listening on the remote host listen on all interfaces, unless the service binds to a specific port by default or if manually configured. In order to show how effective this technique of tunneling over ssh is, we will only allow particular services to listen on the local interface.

You don't have to change your current security configurations, however. We will use tcp_wrappers, that is installed by default with RedHat 7.0 (and previous versions), to connect to the network services. In the `/etc/hosts.deny` file add the following line:

```
ALL : ALL
```

And in your `/etc/hosts.allow` file add the following lines:

```
sshd : ALL
in.ftpd : 127.0.0.1
ipop3d : 127.0.0.1
imapd : 127.0.0.1
```

This sets sshd (the ssh server) to allow connections from anywhere any IP address. The other services only allow connections from the local interface. You can verify this by configuring a mail client to connect to your remote pop or imap server and/or an ftp client to connect to your ftp server, right now. It won't allow you to connect. You'll also need to set up any user accounts to allow access to these services. (Note: The setup above is only useful if the services are only for internal use and remote users need to access the internal services to send and receive email or transfer files. The services can be available for public use and be encrypted with ssh and MindTerm.) If MindTerm will be used over the web to create tunnels or use the secure copy GUI features then a Java Runtime Environment (JRE) will need to be installed on the server running SSH as well.

4.2. Client Configuration

The only client configuration that is needed is to be sure that a JRE is installed for your platform. Windows and MacOS 8 and later have a JRE already installed. It is recommended to install Sun's JRE on Windows. IBM has a list of ports of JRE's to various plaforms: http://www−105.ibm.com/developerworks/tools.nsf/dw/java−devkits−byname as well as Sun: http://java.sun.com/cgi−bin/java−ports.cgi. (You don't need the entire Java package with the debuggers and compilers you just need the Java Virtual Machine to run java applications.) Also, for the tutorial that follows, unzip the MindTerm archive, MindBright's or ISNetwork's implementation, archive into `c:\mindterm` for windows.

---

5. Creating the tunnels

MindTerm can be started a few ways. If you have the JRE installed then you can double−click on the mindtermfull.jar application file. Another way is to open up a dos−shell and type the command:

```
jview −cp c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
```

or

```
javaw −cp  c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
```

or

```
java −cp c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
```

(jview is used if you are using Windows and you don't download the JRE. Javaw comes with the Windows JRE download and is used because a dos−shell box won't be needed in order to run MindTerm so there is one less window open)

MindTerm 2.0 is now available. The argument to start it has changed slightly. Instead of the command above:

```
java −cp c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
```

this will start MindTerm from the commandline:

```
java −cp c:\mindterm\mindtermfull.jar com.mindbright.application.MindTerm
```

Only the "com." was added to the applet parameter.

This will start the MindTerm program and you can then type the server name when prompted and it will prompt you to " Save as Alias". You can type a short server name so when you start the applet again you can simply type the **Alias** you created. You will then be prompted for your login name. After you type it, hit enter and a dialog box will appear informing you that the host doesn't exist and prompt you to create it. Click **Yes**. Another dialog will appear prompting you if you want to add that host to your `known_host` file. Click **Yes**. Then you are prompted for your password. Type your password and hit enter. If you supplied the proper username and password then you should be at a command line on the server you specified.

We'll create a tunnel to the POP and SMTP server, first. After you have successfully logged in (and optionally enabled vlock) click on Tunnels on the menu and then click Basic. A dialog box will appear. Add the following settings to each box, respectively:

- Local port: **2010**
- Remote Hosts: *Your remote host (this should be the server running the sshd server).*
- Remote port: **110**

Now click **Add**. A dialog box should appear stating "The tunnel is now open and operational". *(Note: If you select a port that is already open an error message will appear stating " Could not open tunnel. Error creating tunnel. Error setting up local forward on port XXXX, Address in use.)* Click **OK** and the tunnel configuration should appear in the box now. Click **Close Dialog**. Open up your email client's options or preferences menu. We'll use Netscape Messenger for this example.

1. Open up Netscape
2. Click on **Edit –> Preferences**.
3. On the left column click on **Mail " Newsgroups**, if the contents aren't already displayed.
4. Click on **Identity** and type your information in each box.
5. Click on **Mail Servers** in the left column. The default install of Netscape has "mail" in the box underneath Incoming mail servers.
6. Click on **mail**.
7. Click **Edit** to the right of that box and a dialog box should appear.
8. If POP is not already selected in that drop down box, select it now.
9. In the Server Name box type **localhost:2010** *(remember we chose that local port in the MindTerm tunnel creation menu to forward to the remote servers POP (110) port)* and then your username. Set any other options as you see fit.
10. Click **OK**.
11. In the box **Outgoing mail (SMTP) server** type your smtp server name and underneath that type your Outgoing mail server user name.
12. Click **OK**. *(Don't do anything to the Use Secure Socket Layer (SSL) or TLS for outgoing messages option).*
13. Now click on **Communicator** on the menu.
14. Click **Messenger**.
15. You should then be prompted for your password. Type your password and hit enter. If you have mail you should now be able to read it.

As long as you have a MindTerm ssh session open, this should work with most email clients. Remember that the remote server name or POP server name will be "*localhost:*". If you are asked for the POP server and port seperately then add it accordingly. Any connections to the local port 2010, in this example, will be forwarded to the remote hosts' port 110. If you configure an ftp client to connect to the localhost port 2010, right now it wouldn't work. Why? The POP protocol doesn't understand ftp protocol. Only POP clients can be forwarded to the localhost port 2010 for the tunnel to be effective. A POP server isn't any good if you don't have an smtp server. If you have a mail program like Postfix (www.postfix.net), Qmail (www.qmail.org), or Sendmail (www.sendmail.org) then a secure tunnel can be created to it, as well.

With the MindTerm client still running click on Tunnels again then Basic and add these settings.

- Local Port: **2025***(just type over the settings set from what we did previously)*
- Remote Host: *Your remote smtp server.*
- Remote Port: **25**

Click **Add**. Then click **OK** on the confirmation menu. Now smtp should be added to the list underneath the settings for POP. In the Netscape Messenger mail server settings add: **localhost:2025** as your *Outgoing mail (SMTP) server*. All email you send to the remote host will be encrypted. However, if you send mail to someone outside of the remote host's mail server, your email will be encrypted only from your local machine to your remote smtp server. From the remote smtp server to any other host, will not be encrypted, unless you've configured a tunnel to the other hosts.

To enable encrypted ftp sessions add these settings to a new tunnel.

- Local Port: **2021** *(just type over the settings set from what we did previously)*
- Remote Host: *Your remote ftp server.*
- Remote Port: **21**

Click **Add**. Then click **OK** on the confirmation menu. Now ftp (see the leech ftp example and wsftp−− picture 1 and picture 2) should be added to the list underneath the settings for SMTP.

Imap settings:

- Local Port: **2043** *(just type over the settings set from what we did previously)*
- Remote Host: *Your remote imap server*.
- Remote Port: **143**

Click **Add**. Then click **OK** on the confirmation menu. Now ftp should be added to the list underneath the settings for POP.

All these settings can be automated in a batch file. Simply add the following to a startup script to automatically create a tunnel to your pop server after authentication:

```
jview (or java or javaw) −cp c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
 −server −local0 2010:localhost:110
```

Here is an example based on what we've done above. Add the following to a file in an editor:

```
jview (or java or javaw) −cp c:\mindterm\mindtermfull.jar mindbright.application.MindTerm
 −server −local0 2010:localhost:110 −local1 2025:localhost:25 −local2 /ftp/2021:localhost:21
−local3 2043:localhost:143
```

now save it with a `.bat` extension. Double−click on it. You should be prompted for your login name when MindTerm starts up then type your password. After you are authenticated click on the **Tunnels** menu and click **Basic**. You should see the tunnels in the box that opens up. This is an easy way to allow remote users to start up the tunnels without many configurations on their part. They only need to click the `.bat` file and type their username and password and optionally run vlock. Their client software can be pre−configured for remote profiles that connect to the tunnels automatically.

When you are finished using the MindTerm, be sure to close all applications that are using a tunnel. If you forget to close the programs using the tunnels, MindTerm will display a message when you attempt to exit from the console or quit the program.

What about VNC and NTOP? These services work the same way. Here the VNC server was running on a RedHat 7.0 workstation. When you start the VNC server, it first listens on port 5901 and each server after that increments up 1 port so the second instance of VNC will listen on port 5902, and the third 5903, etc.. On Linux, you can run multiple VNC servers and people can connect to each VNC server as well. In MindTerm you can simply add a VNC tunnel with the following settings:

- Local Port: **2001**
- Remote Host: *Your remote VNC server host name*.
- Remote Port: **5901** *(If this is the first server instance running)*

Click **Add**. Then click **OK** on the confirmation menu.

Run the vncviewer application on your local machine and type: **localhost:2001**, and then the password, when prompted, for the VNC desktop and you have an encrypted VNC session.

Ntop works the same way. If you want to run ntop in web mode as a network monitor, you can tunnel connections to your local machine and view the stats in your local browser, without having to install a webserver or opening port 3000 on your remote server. By default, ntop in web mode listens on port 3000 and waits for an http connection to display network stats. Simply create a tunnel to the server running the ssh server and ntop. First run ntop in web mode: ntop −d −w 3000 Then add the settings to the MindTerm tunnel:

- Local Port: **2080**
- Host: *Server running ntop*.
- Remote Port: **3000**

Click **Add**. Then click **OK** on the confirmation menu.

Open up your web browser and in the location bar type: **http://localhost:2080** You should now see the network stats page for ntop (see the ntop man pages to add password protected access to the ntop display). Similarly, if you want to install a web server so you can use web−based applications to control your server or firewall, then just create a tunnel to port 80. You don't have to open up a port on the public interface. Simply bind the webserver to the local interface and create a tunnel to the remote hosts' port 80. For Apache, edit the `httpd.conf` file and change the *BindAddress* * option to **BindAddress 127.0.0.1**. Then add **localhost** to the *ServerName directive*: **ServerName localhost**. Finally, change the *Listen* directive to: **Listen 127.0.0.1:80** As you can see by now MindTerm can secure almost any TCP service. It can be used on a remote server to run Webmin, which is an excellent web−application to administer your servers. It comes with its own perl−based webserver and listens on port 10000 by default. Simply create a tunnel to it using MindTerm and it should work without any changes to the Webmin application or your local web browser. The MindTerm download zip file contains many useful examples, such as using it from the command line and an explanation of all the menu options. MindTerm has more features than outlined in this tutorial but the tunnel option is well worth spending time focusing on.

6. MindTerm over the web

MindTerm can be used over the web as well. Users don't have to download the application. Simply copy the `mindtermfull.jar` file to a directory into a web directory and the users can simply use it as a built–in application or as a stand–alone java applet. For example, create a folder named `mindterm` under your web directory. Copy the `mindtermfull.jar` file, that was used above, into the web directory folder `mindterm`. Then add the file `index.html` to the directory with the following content (snipped from the `README`):

<html> <head></head> <body> <applet archive="mindtermfull.jar" code=mindbright.application.MindTerm width=700 height=400> <param name=server value="<yourserver name>"> <param name=port value="22"> <param name=cipher value="blowfish"> <param name=te value="xterm−color"> </applet> </body> </html>

MindTerm 2.0 is now available. The argument to start the web applet has changed slightly. Instead of the applet parameter above, and the code example below, change the line:

```
<applet archive="mindtermfull.jar"
code=mindbright.application.MindTerm width=700 height=400>
```

to:

```
<applet archive="mindtermfull.jar"
code=com.mindbright.application.MindTerm width=700 height=400>
```

Only the *com.* needs to be added to the applet parameter *code=*. So the code below will be changed to:

```
<applet archive="mindterm_ns.jar" code=com.mindbright.application.MindTerm.class width=1
height=1>
```

Browse to the location of the directory in your web browser *(http://<yourserver name>/mindterm/index.html)*, be sure to have Java enabled in your browser and you should be able to login into the server now.

In order to create tunnels the most recent version of MindTerm has to be downloaded from the MindBright website, version 1.99. That archive contains a signed applet by MindBright that can be used in your web page to create tunnels as explained above. After you have downloaded the latest version, add the `mindterm_ns.jar` file to the `mindterm` directory under your webserver. Now add a file named `standapplet.html` to the `mindterm` directory and add the following code to start MindTerm as a separate client to create tunnels. (*NOTE: The archive contains an applet for both netscape and Explorer*)

<html> <head></head> <body> <applet archive="mindterm_ns.jar" code=mindbright.application.MindTerm.class width=1 height=1> <param name=server value="<yourserver name>"> <param name=port value="22"> <param name=cipher value="blowfish"> <param name=sepframe value="true"><!−− wheter to run in a separate frame or not −−> <param name=autoprops value="both"><!−− enable/disable automatic save/load of settings −−> </applet> </body> </html>

Now browse to the location of the directory in your web browser *(http://<lt;yourserver name>/mindterm/standapplet.html)*. This will start MindTerm as a standalone java applet, the same as if it was started from the commandline. Tunnels can be created using the applet tags so that users don't have to do anything but browse to the page and then login. Then they would access their services just as explained in the above examples. They can, however, create their own tunnels or new tunnels from the *Tunnels* menu as explained above. The `README` that comes with the MindTerm zip archive has many more applet parameters that can be added. As you create tunnels you can then click on **File** and then **Save** so it keeps the tunnels that you have created when you log in again.

A couple of security notes here are you can't connect to another server using the initial login applet. You can only login to the server where the applet is located. However, after you have logged in successfully you can then log in to other servers from the command line. Also, this MindTerm applet is signed by MindBright so you need to contact the sales department at MindBright to obtain a crytographic signature for your organization. That is, if it is needed.

---

7. Security considerations

When an ssh session starts, the public−keys are being sent over an insecure connection until the authentication process is established.. This allows a person to intercept an ssh session and place their own public key in the connection process. SSH is designed to warn the user if a public−key has changed from what exists in their known_host file. The warning that is given is quite noticeable and ssh will drop the connection if the public keys are different, but user's may still trust the certificate because they may think that their company has changed the server's public key. This kind of attack isn't difficult because the dsniff package mentioned earlier contains the tools to perform it. This attack is more commonly called a *"man−in−the−middle attack" (The End of SSL and SSH)*.

A temporary and easy fix for this is to first teach the user's how to recognize the signs that the host key has changed and what to do to get the proper host(s) public key. Second, post the public key for the ssh server(s) on a website, ftp server, or distribute it some other way so that users have access to it at all times.

8. Conclusion

SSH and MindTerm together can provide local and remote users with a high−level of security with a simple and small drop−in application. It can also be used from nearly any platform available. Java was chosen because of its cross−platform compatibility. If there is a JRE available for a platform that someone uses then they can use the MindTerm application to communicate securely over long distances. Since ssh is becoming the standard for remote administration and logins, soon nearly all platforms will be able to run an ssh server. MindBright is currently working on a Java SSH server.

This tutorial also shows how someone can tunnel through a firewall. This is by no means the intention of this paper. It is hoped people will use it for a secure, quick, and free drop−in VPN−like replacement for remote administration, traveling business people, and a hope that other sectors can see the usefulness in this excellent program. As long as you are allowed to make ssh connections then you can tunnel services through to a remote machine. System and Security Administrators should establish policies against tunneling through firewalls because that can cause internal security breaches if used improperly. Remember that the communication is secured but the commands and files that you access and/or download are still being executed on your local and remote machines. Also, any commands you type on most servers are being logged as well. SSH will protect the data over the network or the Internet but what is done on the remote machines can be logged. SSH and MindTerm will not protect against someone gaining access to a remote user's computer and installing key logging programs or other snooping devices.

It is very simple and quick to set up secure communications but the only way to increase the use of secure communication is for users to encourage their company, financial institutions, health care providers, and other businesses to offer secure services.

9. References

Broadband Access to Increase in Workplace. 25 Jan. 2001. CyberAtlas. 12 Mar. 2001 <http://cyberatlas.internet.com/markets/broadband/article/0,,10099_570571,00.html>.

Broadband Moving On Up. 10 Jan. 2001. CyberAtlas. 12 Mar. 2001. <. http://cyberatlas.internet.com/markets/broadband/article/0,,10099_556391,00.html>.

Connolly, P.J. "Secure the home office sensible and easily" Infoworld. 8 Mar. 2001. 22 Mar. 2001. <http://www.infoworld.com/articles/tc/xml/01/03/12/010312tcsoho.xml>.

Eckels, Josh. "Commercial Use" E−mail to Josh Eckels. 13 Mar. 2001

MindTerm: README. MindBright Technology. 3 March 2001 <. http://www.mindbright.se/documentation/README>. Schneier, Bruce. Secrets and Lies: Digital Security in a Networked World. New York:Wiley & Sons, 2000.

Seifried, Kurt. "The End of SSL and SSH" 18 Dec. 2000. SecurityPortal. 12 March 2001 <http://www.securityportal.com/cover/coverstory20001218.html>.

virtual private network: [Definition]. 6 Oct. 2000. Whatis.com. 15 Mar. 2001. <http://whatis.techtarget.com/definitionsSearchResults/1,289878,sid9,00.html?query=virtual+private+network>.

10. Frequently Asked Questions

Nothing yet.