Setting Up a Reverse Tunnel Through a Firewall

The following howto was created when Allstar user W1GHW, Gary, had a need to access an Allstar site that is located behind an unknown number of nat'ed/firewalled networks at the college where he is a professor. I suggested a reverse SSH tunnel and gave him some ideas how to achieve access to the system. He was able to make it work successfully as he has documented below.

The Concept

Directly accessing a protected computer controlling an Allstar linked radio system when that computer is behind a firewall is normally not possible because one purpose of the firewall is to prevent access by unauthorized users. An unsolicited incoming request to access the protected computer has no idea of the computer's local IP address, and the firewall keeps it that way. This situation does not prevent the protected computer from communicating with systems outside the firewall when the request is initiated by the protected computer. The request from the protected computer behind the firewall is packaged with routing information by the firewall which allows responding systems outside the firewall to communicate with the protected computer. Thus only communications initiated by the protected computer behind the firewall gets through the firewall.

One way around the restriction is to have the protected computer initiate a "reverse tunnel connection" to a computer outside the firewall in such a way that the connection remains active over a long period of time. The setup is such that any third computer accessing the external connected computer via a specified port will be able to request a login dialog with the protected computer through this "reverse tunnel" as it is called.

My previous attempt at setting up such a reverse tunnel worked but was quite involved and required that the protected computer periodically check to see if the connection was still active and, if not, to re-establish it. This new method avoids setting up this procedure explicitly and lets the software manage all the infrastructure checking. This new method (new to me) uses a package called autossh.

The Setup Procedure

Please note that references to the "protected" computer refer to the computer behind the NAT or firewall that you would not normally have access to and the external computer is the computer you do have access to.

1.) Download the needed software package:

pacman -Sy autossh

It would also be good to insure that your system is up to date prior to this installation.

2.) In order to initiate the reverse tunnel the protected computer must establish a connection with the external computer. This ordinarily requires an interactive login dialog. Since there is no interactive individual present, a provision must be made for automatic authentication. This is done by building a ssh key that takes the place of the dialog.

On the protected computer behind the NAT or firewall, the Linux command to issue is:

ssh-keygen

This command will prompt you to enter a location and filename where the key will be placed. This is best put in a directory called .ssh (note the dot before the name) located off the directory for root. I suggest you use **/root/.ssh/id_rsa** as the full pathname for the file. (Note that the actual filename, id_rsa, can be anything. Since I had difficulty when I changed the name, I stuck with the suggested name. I am sure I made some other mistake that made it not work.) It will then ask you for a passphrase. Please leave this blank by hitting <enter> only. It will prompt you again and, again, hit <enter> only. This will result in two files being created in the **/root/.ssh/** directory: **id_rsa** and **id_rsa.pub**. These will be copied to the external computer in the next step.

3.) The key just generated must be copied over to the external computer. This step is critical and must be done carefully and accurately! Any deviation in the name or address format will cause the procedure to fail! The command to issue is:

ssh-copy-id -i /root/.ssh/id_rsa -p xxx root@address

(This is in general format) When this command does its copying, it will log into the /root directory so the files copied will be put in the .ssh directory under /root. The filename must be whatever you used during the **ssh-keygen** command in step 2.). The port number used -p xxx should be the ssh port number for the external computer. In Allstar Pi computers, that port is 222 while for other systems port 22 is usually assigned to the ssh function. Also, the root@address must be the same login username and address that you will use to access the external computer. If it is not, the tunnel will refuse the connection request.

During the execution of this command there will be some dialog about authenticity of the external computer's address. An ECDSA key fingerprint will be generated if it does not already exist for the restricted computer and you will be asked to continue (yes/no). You must

type “yes” (as opposed to “y”) to proceed. There will be more dialog about key installation and then you will be prompted for the external computer login password (the root password in my example) which you must provide. This will end the command, and you will be encouraged to try a login by ssh’ing into the external computer. Do this to check for correctness. If all works properly, you should be at the external computer’s prompt without having to enter a password.

4.) Now we establish a temporary tunnel with the following command (Note this command is all on one line! See the script at the end of third article to make it easier to use.):

autossh -M 10984 -o “PubkeyAuthentication=yes” -o “PasswordAuthentication=no” -i /root/.ssh/id_rsa -R 6666:localhost:222 root@address -p XXX

This is a long command and needs some explanation.

- Autossh is the command.
- The **-M** switch sets up the port that will be monitored. In this example that port number is 10984. The port chosen should not be assigned to anything else.
- The **-o** switch specifies that a public key authentication will be used and that a password will not be needed.
- The **-i** switch indicates the location and filename of the ssh key to use.
- The **-R** switch sets up the reverse tunnel. In my example above 6666 is the number of the port on the external computer from where information will be forwarded and 222 is the port on the restricted computer to which information will be forwarded. (Most Allstar Pi systems use port 222 for ssh.) Localhost is an alias for the restricted computer.
- Finally, root@address -p XXX is the exact address of the external address with port number and is necessary. This address must match the one used in the copying procedure outlined in 3.).

When this is completed, you should find yourself logged into the external computer.

5.) At this point the tunnel is established and you should be able to test it. To do so just issue the following command:

ssh -p 6666 root@localhost

If things are working well, you should get a prompt for the password for the restricted computer. (The very first time you issue this command you will be requested to allow the production of an ECDSA fingerprint key for this connection. You should answer “yes”, again.)

Once the password is given, you should be at the prompt for the restricted computer having gone through the external computer through the reverse tunnel. Voila!

6.) The only thing left to do is to establish a background process to keep the tunnel open after you log off the restricted computer. This is done with the following command and should probably be issued after you have rebooted the restricted computer to ensure that the temporary test process is not running (Again, this is entered all on one line!):

autossh -M 10984 -N -f -o “PubkeyAuthentication=yes” -o “PasswordAuthentication=no” -i /root/.ssh/id_rsa -R 6666:localhost:222 root@address -p XXX &

This line would normally be put in the /etc/rc.local file before the line that calls rc.asterisk. This would make it start at each reboot and no manual start would be needed.

Here two additional switches have been added:

- The -N switch suppresses the execution of the command on the external computer. You will not be logged into the external computer when you issue this command.
- The -f switch drops the process into the background.
- The & at the end prevents the command from waiting for a response from the system in the form of an exit code. Without this & the machine might hang up.

Now you can log out of the restricted computer. At anytime later, you can log into the external computer, either directly or via a third computer through ssh, by issuing the command:

ssh -p 6666 root@localhost

After being prompted for the password you will be logging into the restricted machine as before.

Good luck.

<G>