15.6.4 OpenSSH Configuration

Click one of the buttons to take you to that part of the video.

OpenSSH Configuration 0:00-0:42

Let's spend a few minutes discussing how you configure OpenSSH on a Linux system. Before you can use SSH, you first have to install the OpenSSH package on your system.

Be aware that this package is usually installed by default; I've never run across a Linux distribution where it was not, but if for some reason it has not been installed on your system, you can use the appropriate package management command to do so.

For example, you could use yum, apt-get, or Zypper, whichever one is appropriate for your distribution. This package contains both components of SSH. It includes the SSHD daemon, as well as the SSH client.

OpenSSH Configuration 0:43-1:29

The process of configuring OpenSSH involves configuring both the SSH server and the SSH client. You configure the SSHD server daemon using this file right here: /etc/ssh/sshd_config. The SSH client, on the other hand, is configured using either one of these files shown here: /etc/ssh/ssh_config.

This file provides configuration parameters for all users on the system. You can also configure a user-specific file for just your user account by editing the ssh_config file located in the /.ssh hidden directory within your /home directory.

/etc/ssh/sshd_config 1:30-5:38

Let's take a look at the sshd_config file first. Before we go on, please make sure that you are editing the right version of the file. I have made this mistake many times myself, and I've seen other people make this same mistake. If you're going to edit the server, you need to look at the sshd_config, not the ssh_config.

If you're going to work on the client, then that's the ssh_config file.

When you're opening a file with, say, the vi editor, it is really easy to end up editing the wrong version of the file and then wonder why nothing works correctly. Working on the server? Work on sshd_config.

There are many directories within this file that you can modify. My experience has been, however, that after you install the OpenSSH package, the default parameters within this file usually work well in just about every situation.

To initially get SSHD up and running, you really shouldn't have to make any changes to this file at all. After you get everything working, you may want to customize it for your particular implementation, and there are several different parameters that you can use to do that.

Some of the more useful ones are shown here. The first one is AllowUsers. This restricts logins to the SSH server to only those users listed in this parameter. You can specify a whole list of different users with this parameter. You just need to make sure they are separated by spaces.

DenyUsers prevents the users listed from being able to log in to the SSH server. And again, you specify a list of users separated by spaces. Here's the key thing that you need to remember about these two parameters. You need to use one or the other.

If you specify AllowUsers, then only the users you specify will be allowed to access the system through the SSHD daemon, anybody else will not. Basically, everyone is denied except for those with this parameter.

If you use this parameter, on the other hand, everyone is allowed to access the system through the SSHD daemon, except for those listed in this file.

The next parameter is the HostKey parameter, which specifies which private host key should be used by SSH, because you could create multiple key pairs on your server system; and if this is the case, then you need to specify which one the SSHD daemon should use.

For example, you could configure it to use the RSA key pair, or you could configure it to use the DSA key pair.

One thing that you need to keep in mind, however--and this kind of a little gotcha--if the key file that you specified here with HostKey has read or write permissions assigned to it, owner is okay, but assigning read or write to group or others will cause SSHD to say, "Ah, not gonna use that key." That can cause problems, so make sure that only owner has read or write permissions to the key file that you want to use.

Next we have the ListenAddress parameter. This is useful only in situations where the system has multiple network interfaces installed. If you want to restrict the SSHD daemon to listening only on a particular network interface, then you would specify its IP address with ListenAddress.

You can also use PermitRouteLogin. This parameter specifies whether or not the root user is allowed to access the system directly through the SSHD daemon. On most distributions that I've worked with, this is always turned off.

You're not allowed to log in directly as root. You are able to log in as a standard user and then use the su command at the shell prompt to switch to root. No problem there, it's just that you can't log in to SSHD directly as root.

The next one is Port. This specifies the port on which the SSHD daemon will listen for SSH client requests. By default, this is port 22.

And then finally we have Protocol, which specifies which version of SSH you want to use. You can configure it to use version 1 or you can set it to version 2. Or, you can support both versions, which you probably shouldn't have to do anymore, by entering '2,1', which will give preference to version two, but will still allow version one connections if necessary.

Client Configuration Precedence 5:39-6:43

The SSH client, on the other hand, is configured using a different set of files, and we looked at this earlier. It's the /etc/ssh_sshconfig file and the ssh_config file located in the .ssh hidden directory in the user's /home directory.

As we talked about a minute ago, this file is used to specify default parameters for all users running the SSH client on the system. This one is used to override these defaults, using user-specific configuration parameters located in just that user's /home directory. The precedence for determining which parameters to use for an SSH client session are shown here.

First of all, any command line options that are included with the ssh command at the shell prompt will be given first precedence. The next order of precedence is files located in the ssh_config file in your /home directory. The lowest level of precedence are the settings in the ssh_config file located in the /etc/ssh directory.

Client Configuration Options 6:44-9:34

As with the SSHD daemon, the default parameters used in the ssh_config file for the client usually work just fine without a whole lot of customization. There are parameters in there, though, that you can modify if you wish. Some of the more useful ones are shown here.

For example, you can specify Port if, for some reason, your SSHD server system is using a different port than the default of 22. That's actually a security measure that a lot of system administrators employ, because if I run a port scan against the system as an attacker to see what's available and I see port 22 open, what do I know automatically? That SSH is running.

If, however, I were to go to the SSHD daemon configuration file and change it to some weird port that's not used by anything else, like 7631, and I see that port open on the system, as an attacker I have no idea what that port is used for. It provides a little bit of security through what we call obscurity.

If you do this, however, you do have to go into your clients and say, "Hey, don't try to connect over port 22. Use this weird, funky port instead."

You can also configure which protocol to use, as we saw with the configuration file for the SSHD daemon. We can specify either version 1 of SSH, version 2, or both, giving precedence to version 2.

You can also configure strict host key checking. Understand that the SSH server will send the SSH client its public key whenever you initiate an SSH connection. And by default, the first time you connect to a given SSH server, you will be prompted at the client end to accept that server's public key.

If you want to, you can change this behavior with this parameter right here in the client configuration file. If you set this to a value of Yes, then the client will be allowed to establish connections only to SSH servers whose public key has already been added to the known_host file either the known_host file located in the /.ssh directory in your /home directory, or to the system-wide list of known hosts in /etc/ssh/ssh_knownhost.

If you turn this option on, be aware that if you then try to connect to a new SSH server with the SSH client, it's not going to work. You're not going to be given the option of adding the SSH server's public key to your list of known hosts.

Instead, you are going to have to manually do it first. Once you have done that, then you'll be able to connect to the SSH server. It increases the security of the system, but also makes it a little bit harder to use. You can also specify User, which allows you to specify which user account you're going to connect to the remote SSH server as.

Connecting with the OpenSSH Client 9:35-11:50

Whichever port you decide to use between your client and your server, please be sure that you open up that port in the host-based firewall on the SSH server system, where the SSHD daemon is running. Otherwise, you're not going to be able to establish connections with that system.

Then after you configure your firewall, you can go ahead and connect with the client on your local computer to the remote Linux system running the SSHD daemon, using the command shown here. We enter 'ssh -l', followed by the username on the remote system that you want to connect as, followed by the hostname or IP address of the server you want to connect to.

This is a point of confusion.

Remember that when we're logging in to the remote system, we have to specify the name of a user on the remote system--not on the local system-- but on the remote system that you want to connect as.

If you forget, and leave this parameter out, then the SSH client will automatically try to connect you to the remote system as whoever you are currently logged in to the local system as. And in my experience, that rarely works, because most likely you have a different set of user accounts on the local system as you do on the remote system.

An example of connecting is shown here, where I use the ssh command to log in to a remote system with a hostname of fs5.corpnet.com, and I connect as my rtracy user account.

When that happens, you'll notice right here that we get a warning message saying, basically, "I don't have the public key for the system you're trying to log in to. Do you want me to add it, yes or no?" And I say, "Yep, please add it."

At which point, notice that the remote system is added to the list of known hosts, which is a fancy way of saying that we've grabbed the public key from the remote server, and we've added it to our known host file. I provide my password, and I'm authenticated to the remote system as the user specified.

Notice, when I do, that I'm actually now working at the prompt of the fs5 system, not at the prompt of the system that I ran the SSH client from. I'm essentially remotely accessing that system and performing tasks on it. At this point, any commands I type at the prompt will be actually sent over to the remote system and they will be processed there.

When you're done and want to close the connection, all you have to do is type 'exit', and it will close the SSH connection.

Summary 11:51-12:04

That's it for this lesson. In this lesson we reviewed how to configure OpenSSH on a Linux system.

We first talked about how to configure the SSHD daemon, the server half of OpenSSH. Then we talked about how to configure the SSH client. And then we ended this lesson by talking about how to connect to an SSH server, using the SSH client.