15.6.6 OpenSSH Configuration Facts

Open Secure Shell (OpenSSH) is a suite of utilities that provides secure network connections for remote login and file transfer. OpenSSH uses the SSH protocol for authentication and encryption.

This lesson covers the following topics:

- Configuration files
- Commands used SSH

Configuration Files

OpenSSH uses the following configuration files:

File	Description	Examples
/etc/ssh/sshd_config	The **/etc/ssh/sshd_config** file configures the SSH daemon on the server system. Be aware of the following important options for configuring an SSH server: • **AllowUsers** lists users allowed to use SSH. If an **AllowUsers** line is used in the file, all users except those listed are denied access by default. • **DenyUsers** lists users not allowed to use SSH. If a **DenyUsers** line is used in the file, all users except those listed are granted access by default. • **Protocol** specifies which version of SSH is allowed when establishing a connection to a server. ◦ SSH 2 is more secure, and is supported on newer distributions. The default is protocol 2. ◦ SSH 1 is typically used on older systems. • **ListenAddress** specifies the addresses that SSH should use when listening for requests. By default, the server listens on all IP addresses assigned to it. Use this line to specify specific addresses. • **Port** specifies the port number. The default is 22. Use this line to change the default.	**AllowUsers jsmith mkimball** grants access to the specified users. **DenyUsers gedwards fjones** denies access to the specified users. **Protocol 1,2** allows the server to use either protocol. **ListenAddress 192.168.10.10:22** ensures that the server listen only on the IP address specified. **port 11111** changes the SSH port from the default port to port 11111.

	• **UsePAM** enables the Pluggable Authentication Modules (PAM) interface between sshd and the system. • **PermitRootLogin** specifies whether users can log in as root over SSH. If set to **no**, then users must log in using standard user credentials, but they can use the **su** command to elevate their privileges. Restart the sshd daemon to implement changes.	
/etc/ssh/ssh_config **~/.ssh/config**	The **/etc/ssh/ssh_config** file configures OpenSSH for all users on the client system. **~/.ssh/config** is a user-specific hidden file which can override the configuration settings in **/etc/ssh/ssh_config** file. Both files can be overridden using command line options included with the **ssh** command. Be aware of the following important options for configuring an SSH client: • **Protocol** specifies which version of SSH is allowed when accessing the SSH server. • **StrictHostKeyChecking** determines whether SSH clients can accept keys from SSH servers not previously accessed. The keys of previously accessed servers reside in one of the following directories: ○ **/etc/ssh/ssh_known_hosts** ○ **~/.ssh/known_hosts** If this parameter is set to **yes**, then new keys must be added manually using: **cat *keyfile*.pub >> /etc/ssh/ssh_known_hosts** • **CheckHostIP** verifies that the supplied key matches the IP address of the server when set to yes. This prevents IP spoofing, but might generate warnings or refuse connection if the server changes its keys. • **Port** specifies the port used to connect to the SSH server. • **User** automatically logs in with the specified user name instead of requesting a username.	**Protocol 1,2** allows the client to connect using either protocol. **StrictHostKeyChecking no** automatically adds new host keys to the known hosts files. Other options are *yes* and *ask*. *Ask* is the default. **port 11111** changes the SSH port from the default port to port 11111. **user jsmith** automatically uses jsmith as the user when logging in.

Commands Used with SSH

The following table lists commands used in conjunction with SSH:

Command	Function	Examples
service sshd	Manages the current state of the SSH daemon on the server for init-based distributions. Options include: • **start** starts the sshd daemon if it is not currently running. • **stop** halts a running sshd daemon. • **restart** stops and restarts the sshd daemon. • **status** shows the status of the sshd daemon. ⓘ As an alternative method, use the absolute path to the daemon script and the option to configure the daemon (e.g., **/etc/rc.d/init.d/sshd start**).	**service sshd start** starts the ssh daemon. **/etc/init.d/sshd start** starts the ssh daemon. **service sshd status** shows whether the sshd daemon is running. **/etc/it.d/sshd status** shows whether the sshd daemon is running.
systemctl ***command*** **sshd**	Manages the current state of the SSH daemon on the server for systemd-based distributions. Commands include: • **start** starts the sshd daemon if it is not currently running. • **stop** halts a running sshd daemon. • **restart** stops and restarts the sshd daemon.	**systemctl start sshd** starts the sshd daemon. **systemctl stop sshd** stops the sshd daemon. **systemctl restart sshd** restarts the sshd daemon. **systemctl status sshd** displays the status of the sshd daemon.

	• **status** shows the status of the sshd daemon.	
ssh	Makes a secure connection from the SSH client to the SSH server. Options include: • **-l** specifies the user account on the server. Without this option, SSH uses the same user account name being used on the client computer. • **hostname** specifies the SSH server hostname. Alternatively, the server's IP address could be used. • **commands** (optional) runs the command on the remote system.	**ssh -l bjones hs1** logs in to the hs1 computer as bjones. **ssh -l bjones@hs1 ls -la /var/log** logs in to the hs1 computer as bjones, executes the **ls -la /var/log** command on the remote computer, and then closes the connection.
scp	Encrypts and copy files from a remote system over the network. Options include: • ***username*** specifies the user account on the remote system. • **@hostname:remotefile** specifies the remote system and the remote file. • ***destination/filename*** specifies where to locate and what to name the new file. If the new file name is omitted, the file is copied using the original name.	**scp bjones@hs1:hostfile ~/** copies hostfile from the server to the home directory on the client. **scp ~/clientfile bjones@hs2.mydomain.com:/home/bjones**/ copies clientfile from the client computer to the home directory of bjones on the server. **scp bjones@hs1.corpnet.com:/home/bjones/file1 bjones@hs2.corpnet.com:/home/bjones/** copies file1 from hs1 to hs2.
sftp	Transfers files securely from ftp servers. Options include: • ***username*** specifies the user account on the remote system. • **@hostname** specifies the remote system. After logging in, use the same commands that are used when	**sftp bjones@ftp1.corpnet.com** connects to the FTP server as bjones.

	using **ftp**. These include: • **pwd** shows the current directory on the sftp server. • **get** ***file_name*** copies a file from the sftp server. • **put** ***file_name*** copies a file to the sftp server. • **cdup** traverses up a directory. • **ls** displays files in the current directory on the sftp server.	
slogin	Allows access to a shell on a remote computer. It is identical to the **ssh** command. This is supported only for backwards compatibility.	**slogin -l bjones hs1** logs in to the hs1 computer as bjones.