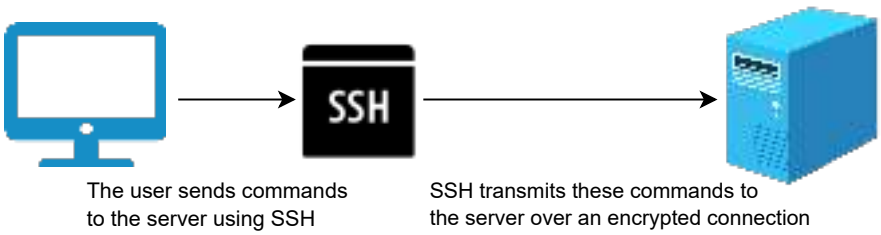
Working with Secure Shell

In this lesson, we will learn how to communicate with our server remotely.

WE'LL COVER THE FOLLOWING

Secure Shell, popularly known as **SSH**, is a platform which allows us to remotely control our server over an internet connection. It is a terminal based protocol system which is available on macOS, Linux and Windows.

SSH was originally developed because of the security concerns regarding its predecessor, **Telnet**. Server communication is now much more secure because of the data encryption methods used by the SSH protocol. In terms of functional power, SSH allows us to basically control every aspect of the server running our website.



Now, let's take a look at how we can set up the SSH terminal on our machine.

The SSH Environment

SSH comes in-built with Linux and macOS. We can access it using the terminal. For Windows, there are several SSH clients like **PuTTY** which allow

us to work with SSH.

Basic Commands

The most important SSH command is to open a connection to our remote server. Here is the command we need to run on the terminal:

```
<user>@<domain name or IP address>
```

First, we need to define the `user`, which is the account we want to use. The next step is to mention the domain name or the IP address of our website. For example, the root user would open up a connection on our platform this way:

```
root@educative.io
```

Or we could use the IP address like this:

```
root@192.1.1.1
```

We must then put in our password to access the server. This connection is secured by SSH using **hashing**, **symmetric encryption** and **asymmetric encryption**. Without getting into details, all we need to know is that this is a big improvement over Telnet's security measures.

Below, we've listed a few more basic operations that can be performed on the SSH terminal.

Command	Purpose	Syntax
ls	Display all the files and directories on the server machine.	`ls` (basic) or `ls -a` (shows hidden files as well)
mkdir	Create a new directory.	`mkdir newDirectory`
cd	Move into the specified directory.	`cd anyFolder`

touch	Create a new file with a specified format	`touch newDoc.txt`
rm	Delete a file or directory. Use `rm -r` to remove all files in the directory being deleted	`rm newDoc.txt`
pwd	Print the current directory we are working in	`pwd`
cp	Copy files and directories.	`cp source destination` (we can use several flags before `source`)
scp	Copy files/folders to and from the server.	`scp filename host:path` (`host` can be the host domain name or the IP address)
mv	Move files and directories.	`mv source dest`
find	Search for a file in a given directory. It can also return multiple files.	`find . -name file.txt`

This list is not exhaustive by any means. For more details, visit Secure Shell's official website.

Port Number

By default, SSH connects to **port 22**, but we can always choose a different port if we want. Opting for another port is also useful because hackers will only try to attack port 22.

To change the port number for SSH, we run the following command:

```
ssh -p port user@server
```

Git Repositories

If Git is installed on our machine as well as the server machine, we can create private Git repositories using SSH. We simply have to move to the desired directory on the server and run this command:

```
git init
```

Now that we've understood SSH, we'll move on to the File Transfer Protocol and learn how it is incorporated with SSH.