## How to Install and Configure SSH on Ubuntu 22.04

The primary step in setting up remote access to your Ubuntu machine is to enable SSH. SSH, which stands for Secure Shell, is a network protocol that enables users to securely connect to remote resources over an unprotected network. This tool is indispensable for system administrators, developers, and anyone who needs to manage remote connections and transfer files safely. By leveraging SSH, you can confidently connect to your Ubuntu machine from another device, as all communications are secured through encryption, safeguarding against data theft and lessening the risk of remote network attacks.

Consider that you have obtained a cloud server from Hostman. To facilitate connection and administration of the server, SSH must be installed and configured. This guide will walk you through the installation and configuration process of SSH on Ubuntu 22.04.

**Step 1: Prepare Ubuntu**

Prior to installing SSH on Ubuntu, it is important to make sure that all the appropriate apt packages

are updated to their latest versions. This can be accomplished by executing the given command:

**sudo apt update && sudo apt upgrade**

**Step 2: Install SSH on Ubuntu**

In most cases, the SSH server is included as a standard feature in your Ubuntu system. If the OpenSSH is not included by default, manual installation will be required. To proceed with the installation, type the command indicated here into your terminal.

**sudo apt install openssh-server**

As you install, be sure to select "Yes" for all the system prompts that appear. When the installation is done, go ahead and proceed to the next step to activate the service.

We must first confirm whether our OpenSSH Server is installed on our system. Carry out this command to move forward.

**sudo apt list openssh-server**

## **Step 3: Start SSH**

To initiate the SSH service which is recently installed, make use of the subsequent instructions given below:

If you want to kick off the SSH service which is recently just set up, check out the steps listed below.

**sudo systemctl enable --now ssh**

The inclusion of the **--now** flag will the service and configure it to start automatically when the system powers up.

To determine the operational status of the service, utilize the command indicated below:

To see if the service is up and running, make sure to apply the command listed here:

**sudo systemctl status ssh**

In the output, find the line that reads **Active: active (running);** this signifies that the service is operating without issues.

If you intend to deactivate the service, simply apply the command listed here:

**sudo systemctl disable ssh**

This procedure will terminate the service and block it from activating upon system startup.

**Step 4: Configure the Firewall**

Prior to establishing an SSH connection to the server, it's important to check that the firewall settings are set up correctly. If a firewall is actively running, it is necessary to permit SSH traffic. For those employing UFW (Uncomplicated Firewall), you can enable SSH access by typing in this command.

**sudo ufw allow ssh**

Also, make sure that UFW is configured to start automatically every time the system starts.

**sudo ufw enable**

You can assess its current status by performing the command mentioned below as you are operating with UFW.

**sudo ufw status**

If SSH traffic is blocked, you are able to authorize it by using the command detailed below. then follow the previously mentioned steps for setup.

**sudo ufw allow ssh**

and follow the above-mentioned steps for configuration.

## **Step 5: Connect to the Server**

With the earlier steps behind you, it's time to log into the server utilizing the SSH protocol. You'll need the server's IP address on the other hand the domain name, plus the username of a user that exists on the server. Just type the command into your terminal to get started.

**ssh username@IP\_address**

Alternatively:

**ssh username@domain**

**Important**: To successfully connect to a remote server, SSH must be installed and configured on the remote server and the user's computer from which you make the connection.

To verify whether SSH is functioning, you can connect to your server from a different machine with the running the command:

**ssh your\_username@your\_server\_ip\_address**

Substitute your\_username with your server username and your\_server\_ip\_address with the corresponding IP address of your server.

## **Step 6: Configure SSH**

Now that you've perfectly navigated the first five steps, you're ready to establish a remote connection to the server. However, to enhance the security of this connection, consider altering the default connection port or switching from password authentication to key-based authentication. Making these adjustments will require modifications to the SSH configuration file.

The chief settings for the OpenSSH server are contained within the **sshd\_config** file, which can be stored at **/etc/ssh**.

### Prior to making any changes, it’s important to back up this file.

To safeguard your work, consider duplicating this file prior to making any modifications.

**configuration.sudo cp /etc/ssh/sshd\_config /etc/ssh/sshd\_config.initial**

This precaution will enable you to restore the original version if any mistakes happen after your changes.

### **Edit the Configuration File**

To alter the configuration file, you may conveniently open it with the nano text editor.

**sudo nano /etc/ssh/sshd\_config**

#### **Change the Port**

To modify the configuration file, you can easily open it using the nano text editor. Within the file, adjust the port to a more secure option. It is advisable to select values from the dynamic port range of 49152 to 65535 and to use varied numbers for increased security. For instance, we can set the port value to 48582. This can be accomplished by uncommenting the relevant line in the file and tweak the port number.

#### **Enable Key Authentication**

Moreover, it is recommended to modify the password authentication setting to a more secure key authentication mode. This requires uncommenting the corresponding line and verifying that the value is "**Yes**,"

**PasswordAuthentication no**

To

**PasswordAuthentication Yes**

#### 

#### **Disable Root Login**

It is now necessary to implement a prohibition on superuser access to the server by adjusting the corresponding line by setting the following value.

**PermitRootLogin no**

## **Additional Security Settings**

 Below are some recommended configurations that can be implemented to improve the server's security

● **UseDNS**: Verifies whether the hostname corresponds to its associated IP address. The designation "Yes" activates this parameter.

● **PermitEmptyPasswords**: If configured to "No," this option blocks the use of empty passwords during the authentication purposes.

● **MaxAuthTries**: establishes a cap on the number of failed attempts to connect to the server during a single communication session.

● **AllowUsers and AllowGroups** : manage the roster of users and groups that have permission to access the server via SSH.

**# AllowUsers User1, User2, User3**

**# AllowGroups Group1, Group2, Group3**

● **Login GraceTime**: parameter determines the duration allowed for successful user authorization. It is advisable to decrease this value to one-fourth of its current setting.

● **ClientAliveInterval**: setting controls the maximum period of user inactivity. When a user goes beyond this limit, they'll lose their connection.

Once you've made all the changes to the major configuration file, make sure to save your work and exit the editor.

### **Restart the SSH Service**

To put this modification into action, you have to kickoffthe SSH service again.

**sudo systemctl restart ssh**

In the event that the port number has been modified, please connect using the updated port as follows:

**ssh -p port\_number username@IP\_address**

Alternatively, you may use:

**ssh -p port\_number username@domain**

**Conclusion**

Here’s a comprehensive guide that walks you through the installation and configuration process of SSH on Ubuntu 22.04. It also includes steps to adjust the main configuration file for better security. We anticipate this guide makes it easier for you to establish a secure remote connection to your Ubuntu server.