

[Subscribe](#) [Share](#) [Contents](#) ▼

Initial Server Setup with Debian 9

Posted September 4, 2018 102.1k

GETTING STARTED

INITIAL SERVER SETUP

SECURITY

DEBIAN 9

DEBIAN



By: Justin Ellingwood

Not using **Debian 9**? Choose a different version:

Automated: Bash



CentOS 7



Debian 8



Introduction

When you first create a new Debian 9 server, there are a few configuration steps that you should take early on as part of the basic setup. This will increase the security and usability of your server and will give you a solid foundation for subsequent actions.

Step One — Logging in as Root

To log into your server, you will need to know your **server's public IP address**. You will also need the password or, if you installed an SSH key for authentication, the private key for the **root** user's account. If you have not already logged into your server, you may want to follow our [guide on how to connect to your Droplet with SSH](#), which covers this process in detail.

If you are not already connected to your server, go ahead and log in as the **root** user using the following command (substitute the highlighted portion of the command with your server's public IP address):

```
$ ssh root@your_server_ip
```

Accept the warning about host authenticity if it appears. If you are using password authentication, provide your **root** password to log in. If you are using an SSH key that is passphrase protected, you may be

prompted to enter the passphrase the first time you use the key each session. If this is your first time logging into the server with a password, you may also be prompted to change the **root** password.

About Root

The **root** user is the administrative user in a Linux environment that has very broad privileges. Because of the heightened privileges of the **root** account, you are *discouraged* from using it on a regular basis. This is because part of the power inherent with the **root** account is the ability to make very destructive changes, even by accident.

The next step is to set up an alternative user account with a reduced scope of influence for day-to-day work. We'll teach you how to gain increased privileges during the times when you need them.

Step Two — Creating a New User

Once you are logged in as **root**, we're prepared to add the new user account that we will use to log in from now on.

Note: In some environments, a package called **unscd** may be installed by default in order to speed up requests to name servers like LDAP. The most recent version currently available in Debian contains a bug that causes certain commands (like the **adduser** command below) to produce additional output that looks like this:

```
sent invalidate(passwd) request, exiting
sent invalidate(group) request, exiting
```

These messages are harmless, but if you wish to avoid them, it is safe to remove the **unscd** package if you do not plan on using systems like LDAP for user information:

```
# apt remove unscd
```

This example creates a new user called **sammy**, but you should replace it with a username that you like:

```
# adduser sammy
```

You will be asked a few questions, starting with the account password.

Enter a strong password and, optionally, fill in any of the additional information if you would like. This is not required and you can just hit **ENTER** in any field you wish to skip.

Step Three — Granting Administrative Privileges

Now, we have a new user account with regular account privileges. However, we may sometimes need to do administrative tasks.

To avoid having to log out of our normal user and log back in as the **root** account, we can set up what is known as "superuser" or **root** privileges for our normal account. This will allow our normal user to run commands with administrative privileges by putting the word `sudo` before each command.

To add these privileges to our new user, we need to add the new user to the **sudo** group. By default, on Debian 9, users who belong to the **sudo** group are allowed to use the `sudo` command.

As **root**, run this command to add your new user to the **sudo** group (substitute the highlighted word with your new user):

```
# usermod -aG sudo sammy
```

Now, when logged in as your regular user, you can type `sudo` before commands to perform actions with superuser privileges.

Step Four — Setting Up a Basic Firewall

Debian servers can use firewalls to make sure only connections to certain services are allowed. Although the `iptables` firewall is installed by default, Debian does not strongly recommend any specific firewall. In this guide, we will install and use the UFW firewall to help set policies and manage exceptions.

We can use the `apt` package manager to install UFW. Update the local index to retrieve the latest information about available packages and then install the firewall by typing:

```
# apt update
# apt install ufw
```

Note: If your servers are running on DigitalOcean, you can optionally use [DigitalOcean Cloud Firewalls](#) instead of the UFW firewall. We recommend using only one firewall at a time to avoid conflicting rules that may be difficult to debug.

Firewall profiles allow UFW to manage sets of firewall rules for applications by name. Profiles for some common software are bundled with UFW by default and packages can register additional profiles with UFW during the installation process. OpenSSH, the service allowing us to connect to our server now, has a firewall profile that we can use.

You can see this by typing:

```
# ufw app list
```

Output

Available applications:

```
. . .
OpenSSH
. . .
```

We need to make sure that the firewall allows SSH connections so that we can log back in next time. We can allow these connections by typing:

```
# ufw allow OpenSSH
```

Afterwards, we can enable the firewall by typing:

```
# ufw enable
```

Type "y " and press ENTER to proceed. You can see that SSH connections are still allowed by typing:

```
# ufw status
```

Output

Status: active

To	Action	From
--	-----	----
OpenSSH	ALLOW	Anywhere
OpenSSH (v6)	ALLOW	Anywhere (v6)

As the firewall is currently blocking all connections except for SSH, if you install and configure additional services, you will need to adjust the firewall settings to allow acceptable traffic in. You can learn some common UFW operations in [this guide](#).

Step Five — Enabling External Access for Your Regular User

Now that we have a regular user for daily use, we need to make sure we can SSH into the account directly.

Note: Until verifying that you can log in and use `sudo` with your new user, we recommend staying logged in as `root`. This way, if you have problems, you can troubleshoot and make any necessary changes as `root`. If you are using a DigitalOcean Droplet and experience problems with your `root` SSH connection, you can [log into the Droplet using the DigitalOcean Console](#).

The process for configuring SSH access for your new user depends on whether your server's `root` account uses a password or SSH keys for authentication.

If the Root Account Uses Password Authentication

If you logged in to your **root** account *using a password*, then password authentication is enabled for SSH. You can SSH to your new user account by opening up a new terminal session and using SSH with your new username:

```
$ ssh sammy@your_server_ip
```

After entering your regular user's password, you will be logged in. Remember, if you need to run a command with administrative privileges, type `sudo` before it like this:

```
$ sudo command_to_run
```

You will be prompted for your regular user password when using `sudo` for the first time each session (and periodically afterwards).

To enhance your server's security, **we strongly recommend setting up SSH keys instead of using password authentication**. Follow our guide on [setting up SSH keys on Debian 9](#) to learn how to configure key-based authentication.

If the Root Account Uses SSH Key Authentication

If you logged in to your **root** account *using SSH keys*, then password authentication is *disabled* for SSH. You will need to add a copy of your local public key to the new user's `~/.ssh/authorized_keys` file to log in successfully.

Since your public key is already in the **root** account's `~/.ssh/authorized_keys` file on the server, we can copy that file and directory structure to our new user account in our existing session with the `cp` command. Afterwards, we can adjust ownership of the files using the `chown` command.

Make sure to change the highlighted portions of the command below to match your regular user's name:

```
# cp -r ~/.ssh /home/sammy
# chown -R sammy:sammy /home/sammy/.ssh
```

Now, open up a new terminal session and using SSH with your new username:

```
$ ssh sammy@your_server_ip
```

You should be logged in to the new user account without using a password. Remember, if you need to run a command with administrative privileges, type `sudo` before it like this:

```
$ sudo command_to_run
```

You will be prompted for your regular user password when using `sudo` for the first time each session (and periodically afterwards).

Step Six — Completing Optional Configuration

Now that we have a strong baseline configuration, we can consider a few optional steps to make the system more accessible. The following sections cover a few additional tweaks focused on usability.

Installing man Pages

Debian provides extensive manuals for most software in the form of `man` pages. However, the `man` command is not always included by default on minimal installations.

Install the `man-db` package to install the `man` command and the manual databases:

```
$ sudo apt install man-db
```

Now, to view the manual for a component, you can type:

```
$ man command
```

For example, to view the manual for the `top` command, type:

```
$ man top
```

Most packages in the Debian repositories include manual pages as part of their installation.

Changing the Default Editor

Debian offers a wide variety of text editors, some of which are included in the base system. Commands with integrated editor support, like `visudo` and `systemctl edit`, pass text to the `editor` command, which is mapped to the system default editor. Setting the default editor according to your preferences can help you configure your system more easily and avoid frustration.

If your preferred editor is not installed by default, use `apt` to install it first:

```
$ sudo apt install your_preferred_editor
```

Next, you can view the current default and modify the selection using the `update-alternatives` command:

```
$ sudo update-alternatives --config editor
```

The command displays a table of the editors it knows about with a prompt to change the default:

Output

There are 8 choices for the alternative editor (providing /usr/bin/editor).

Selection	Path	Priority	Status

* 0	/usr/bin/joe	70	auto mode
1	/bin/nano	40	manual mode
2	/usr/bin/jmacs	50	manual mode
3	/usr/bin/joe	70	manual mode
4	/usr/bin/jpico	50	manual mode
5	/usr/bin/jstar	50	manual mode
6	/usr/bin/rjoe	25	manual mode
7	/usr/bin/vim.basic	30	manual mode
8	/usr/bin/vim.tiny	15	manual mode

Press <enter> to keep the current choice[*], or type selection number:

The asterisk in the far left column indicates the current selection. To change the default, type the "Selection" number for your preferred editor and press Enter . For example, to use nano as the default editor given the above table, we would choose 1 :

Output

Press <enter> to keep the current choice[*], or type selection number: 1
update-alternatives: using /bin/nano to provide /usr/bin/editor (editor) in manual mode

From now on, your preferred editor will be used by commands like visudo and systemctl edit , or when the editor command is called.

Where To Go From Here?

At this point, you have a solid foundation for your server. You can install any of the software you need on your server now.

By: Justin Ellingwood

 Upvote (9)  Subscribe  Share



We just made it easier for you to deploy faster.

[TRY FREE](#)

Related Tutorials

[How To Install and Secure Memcached on Ubuntu 18.04](#)

[How To Secure a Containerized Node.js Application with Nginx, Let's Encrypt, and Docker Compose](#)

[How to Set Up an Nginx Ingress with Cert-Manager on DigitalOcean Kubernetes](#)



[How To Secure Nginx with NAXSI on Ubuntu 16.04](#)

[How to Get Started with FreeBSD](#)

1 Comment

Leave a comment...

[Log In to Comment](#)

 [rochie94](#) *October 16, 2018*
 0 [get this when using apt-update](#)

E: The repository '<http://security.ubuntu.com/ubuntu stretch-security Release>' does not have a Release file.
N: Updating from such a repository can't be done securely, and is therefore disabled by default.



This work is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.



Copyright © 2019 DigitalOcean™ Inc.

[Community](#) [Tutorials](#) [Questions](#) [Projects](#) [Tags](#) [Newsletter](#) [RSS](#) 

[Distros & One-Click Apps](#) [Terms, Privacy, & Copyright](#) [Security](#) [Report a Bug](#) [Write for DOnations](#) [Shop](#)