Initial Server Setup with Debian 8

Posted May 11, 2015 197.3k GETTING STARTED INITIAL SERVER SETUP LINUX BASICS SECURITY LINUX COMMANDS DEBIAN

By: Mitchell Anicas

Not using **Debian 8**? Choose a different version:

Introduction

When you first create a new Debian 8 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 — Root Login

To log into your server, you will need to know your server's public IP address and the password for the "root" user's account. If you have not already logged into your server, you may want to follow the first tutorial in this series, 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 word with your server's public IP address):

```
local$ ssh root@SERVER_IP_ADDRESS
```

Complete the login process by accepting the warning about host authenticity, if it appears, then providing your root authentication (password or private key). If it is your first time logging into the server, with a password, you will 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 actually *discouraged* from using it

SCROLL TO TOP

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 — Create 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.

This example creates a new user called "demo", but you should replace it with a user name that you like:

```
# adduser demo
```

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 — Root 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 "super user" 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.

Install Sudo

Debian 8 doesn't come with `sudo` installed, so let's install it with apt-get.

First, update the apt package index:

```
# apt-get update
```

Then use this command to install sudo:

```
# apt-get install sudo
```

Now you are able to use the `sudo` and `visudo` commands.

Grant Sudo Privileges

To add these privileges to our new user, we need to add the new user to the "sudo" group. By default, on Debian 8, 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 -a -G sudo demo
```

Now your user can run commands with super user privileges! For more information about how this works, check out this sudoers tutorial.

Step Four — Add Public Key Authentication (Recommended)

The next step in securing your server is to set up public key authentication for your new user. Setting this up will increase the security of your server by requiring a private SSH key to log in.

Generate a Key Pair

If you do not already have an SSH key pair, which consists of a public and private key, you need to generate one. If you already have a key that you want to use, skip to the *Copy the Public Key* step.

To generate a new key pair, enter the following command at the terminal of your **local machine** (ie. your computer):

```
local$ ssh-keygen
```

Assuming your local user is called "localuser", you will see output that looks like the following:

```
ssh-keygen output
Generating public/private rsa key pair.
Enter file in which to save the key (/Users/localuser/.ssh/id_rsa):
```

Hit return to accept this file name and path (or enter a new name).

Next, you will be prompted for a passphrase to secure the key with. You may either enter a passphrase or leave the passphrase blank.

Note: If you leave the passphrase blank, you will be able to use the private key for authentication without entering a passphrase. If you enter a passphrase, you will need both the private key *and* the passphrase to log in. Securing your keys with passphrases is more secure, but both methods have their uses and are more secure than basic password authentication.

This generates a private key, `id_rsa`, and a public key, `id_rsa.pub`, in the `.ssh` directory of the *localuser*'s home directory. Remember that the private key should not be shared with an

not have access to your servers!

Copy the Public Key

After generating an SSH key pair, you will want to copy your public key to your new server. We will cover two easy ways to do this.

Note: The `ssh-copy-id` method will not work on DigitalOcean if an SSH key was selected during Droplet creation. This is because DigitalOcean disables password authentication if an SSH key is present, and the `ssh-copy-id` relies on password authentication to copy the key.

If you are using DigitalOcean and selected an SSH key during Droplet creation, use option 2 instead.

Option 1: Use ssh-copy-id

If your local machine has the `ssh-copy-id` script installed, you can use it to install your public key to any user that you have login credentials for.

Run the `ssh-copy-id` script by specifying the user and IP address of the server that you want to install the key on, like this:

```
local$ ssh-copy-id demo@SERVER_IP_ADDRESS
```

After providing your password at the prompt, your public key will be added to the remote user's `.ssh/authorized_keys` file. The corresponding private key can now be used to log into the server.

Option 2: Manually Install the Key

Assuming you generated an SSH key pair using the previous step, use the following command at the terminal of your **local machine** to print your public key (`id_rsa.pub`):

```
local$ cat ~/.ssh/id_rsa.pub
```

This should print your public SSH key, which should look something like the following:

```
id_rsa.pub contents
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAABAQDBGTO0tsVejssuaYR5R3Y/i73SppJAhme1dH7W2c47d4gOqB4izP0+fRLfvbz/t
```

Select the public key, and copy it to your clipboard.

Add Public Key to New Remote User

To enable the use of SSH key to authenticate as the new remote user, you must add the public key to a special file in the user's home directory.

On the server, as the `root` user, enter the following command to switch to the new user (substitute your own user name):

```
# su - demo
```

Now you will be in your new user's home directory.

Create a new directory called `.ssh` and restrict its permissions with the following commands:

```
$ mkdir .ssh
$ chmod 700 .ssh
```

Now open a file in *.ssh* called `authorized_keys` with a text editor. We will use *nano* to edit the file:

```
$ nano .ssh/authorized_keys
```

Now insert your public key (which should be in your clipboard) by pasting it into the editor.

Hit `CTRL-X` to exit the file, then `Y` to save the changes that you made, then `ENTER` to confirm the file name.

Now restrict the permissions of the *authorized_keys* file with this command:

```
$ chmod 600 .ssh/authorized_keys
```

Type this command *once* to return to the `root` user:

```
$ exit
```

Now you may SSH login as your new user, using the private key as authentication.

To read more about how key authentication works, read this tutorial: How To Configure SSH Key-Based Authentication on a Linux Server.

Step Five — Configure SSH

Now that we have our new account, we can secure our server a little bit by modifying its SSH daemon configuration (the program that allows us to log in remotely) to disallow remote SSH access to the **root** account.

Begin by opening the configuration file with your text editor as root:

```
# nano /etc/ssh/sshd_config
```

Here, we have the option to disable root login through SSH. This is generally a more secure setting since we can now access our server through our normal user account and escalate privileges when necessary.

To disable remote root logins, we need to find the line that looks like this:

/etc/ssh/sshd_config (before)

```
#PermitRootLogin yes
```

You can modify this line to "no" like this if you want to disable root login:

/etc/ssh/sshd_config (after)

```
PermitRootLogin no
```

Disabling remote root login is highly recommended on every server!

When you are finished making your changes, save and close the file using the method we went over earlier (`CTRL-X`, then `Y`, then `ENTER`).

Reload SSH

Now that we have made our changes, we need to restart the SSH service so that it will use our new configuration.

Type this to restart SSH:

```
# systemctl restart ssh
```

Now, before we log out of the server, we should **test** our new configuration. We do not want to disconnect until we can confirm that new connections can be established successfully.

Open a **new** terminal window. In the new window, we need to begin a new connection to our server. This time, instead of using the root account, we want to use the new account that we created.

```
local$ ssh demo@SERVER_IP_ADDRESS
```

You will be prompted for the new user's password that you configured. After that, you will be logged in as your new user.

Remember, if you need to run a command with root privileges, type "sudo" before it like this:

```
$ sudo command_to_run
```

If all is well, you can exit your sessions by typing:

```
$ exit
```

Where To Go From Here?

At this point, you have a solid foundation for your Debian 8 server. You can install any of the software you need on your server now.

By: Mitchell Anicas

Related Tutorials

How to Get Started with FreeBSD

Initial Server Setup with Debian 9

How to Set Up SSH Keys on Debian 9

How to Set Up SSH Keys on Ubuntu 18.04

Automating Initial Server Setup with Ubuntu 18.04

16 Comments

SCROLL TO TOP

Leave a comment...

Log In to Comment

zoresvit *May 12, 2015*

1 Thank you for the post!

However, section about manual copy of the newly generated SSH key is redundant — there is much more efficient way. On your host type:

```ssh-copy-id <server>

  where `<server>` is your server's IP. It will do the rest by itself helping to avoid the hu

Also I wouldn't recommend remapping the default SSH port as it degrades interoperability and

Much better solution would be: set up public key authentication and completely disable passw
edit `nano /etc/ssh/sshd_config` and set:
```PasswordAuthentication no
```

This will eliminate brute force attacks without remapping SSH default port.

manicas MOD *May 12, 2015*

2 Hey there! Thanks for the feedback. We will update the following in the near future:

- Add the `ssh-copy-id` as the primary method (as it doesn't come with OSX, etc)
- Remove the SSH port change (agreed)

Disabling password authentication is covered in another tutorial (that needs to be ported from a different distro).

playahutch *June 7, 2015*

0

SCROLL TO TOP
```

I'm kind of confused. Where is the part about creating the .ppk file to have on your computer to match up with the key on the server? This is my first time using Debian 8 and tutorials are harder to come by, but in every other distro I've used you have to have the .ppk somewhere...

Also, at the beginning we did apt-get update but then never proceeded to do apt-get upgrade. From my understanding (which trust me, is limited) I thought apt-get update downlads the updates but then you have to do apt-get upgrade to actually install the updates?

---

peterwestlake *June 26, 2015*

0 .ppk is the name used for the private key on some systems, notably PuTTY. It probably stands for "PuTTY Private Key", in fact. The equivalent here is the "id_rsa" file mentioned in the instructions.

"apt-get update" updates APT's data about which is the latest release of each package. It's sensible to do that before either installing a new package (as we're doing here) or updating all the already-installed packages (which is what "apt-get upgrade" does).

---

herculesnetwork *August 6, 2015*

0 Thanks for beautiful job here!
Which ports values can I use? I can create any value? eg
54876 or 42976 or 38469 etc ... use my imagination?

I created a crazy door and can log ssh -p NUM*PORT my*user @ my*ip*vps
this because they are accessible and yet did nothing in server .. I shall not have problems with applications?
so to use wordpress

---

manicas MOD *August 6, 2015*

0 Hey there. If you want SSH to listen on a different port, you can specify a new value for the `Port` setting in `/etc/ssh/sshd_config` (e.g. `Port 2222`). You can use any unused port up to 65535.

If you change the port, be sure to restart the SSH service, then specify the port when you connect (e.g. `ssh -p 2222 user@ip`

---

herculesnetwork *August 7, 2015*

0 Thanks Manicas! you help me too ... I already use a different ssh port of standards, have disabled access with root User and changed permissions 700 ~ / .ssh and and 600 to key_file many others things follow their tutorials and am very happy with my super safe wordpress, I used all the safety guidelines found here in the community. ssh use and with different port between 1024 and 54000 is the q values used in my project, and that's ok, we specify yes she with ssh -p MyPort myuser@ myserver and always everything ok in the fast inputs and without login with password ... practical and happy .. but still asked the question because it could have a future problem in using a door any randomly chosen by my imagination .. use 5 digits on my door .. but now that I know there are no problems, since so well tranquil . THANK YOU AGAIN! success for you.

---

adamASDF *December 18, 2015*

SCROLL TO TOP

1 Great tutorial. Worked like a charm.

I'm guessing most people that would be attempting this would already know this but just in case...

`ssh-copy-id` doesn't ship with OSX but if you have Homebrew installed (you should, or equivalent package management software, if not you probably shouldn't be attempting this), you can install it with:

```
brew install ssh-copy-id
```

dannyid *January 2, 2016*

0

Hi. Thank you for the awesome tutorial. It helped beautifully. The only part that didn't work for me was Step Five and turning off root ssh access.

I added `PermitRootLogin no` to my `sshd_config` file (it was *not* there before with a `yes` value) and restarted `ssh` but that didn't prevent root ssh access.

I noticed in my `sshd_config` file there is a section that states:

```
# Set this to 'yes' to enable PAM authentication, account processing,
# and session processing. If this is enabled, PAM authentication will
# be allowed through the ChallengeResponseAuthentication and
# PasswordAuthentication.  Depending on your PAM configuration,
# PAM authentication via ChallengeResponseAuthentication may bypass
# the setting of "PermitRootLogin yes
# If you just want the PAM account and session checks to run without
# PAM authentication, then enable this but set PasswordAuthentication
# and ChallengeResponseAuthentication to 'no'.
UsePAM yes
```

Any ideas how to turn off root ssh with PAM?

mindyk *January 14, 2016*

0

```
systemctl restart ssh
```

did not work for me , but this did

```
/etc/init.d/ssh restart
```

luizsantosws *March 30, 2016*

0

PuTTY Fatal Error
Couldn't agree a client-to-server cipher (availble: aes128-ctr,aes192-ctr,aes256-ctr,aes128-gcm@openssh.com.aes256-gcm@openssh.com, chacha20-poly1305@openssh.com

mattdelac *August 28, 2016*

0 Hi everyone,

This would be great if you have time to give back comment about a set-up script I just wrote:
https://github.com/MattDelac/set_up_security_server/

All comments are welcome
Thanks

shwelu *May 4, 2017*

0 Thank you for the post!

At present, it seems that login using password is disabled by default. Password sent to the mail on creating a droplet for the first time --but not adding SSH key before creating the droplet, is good only for changing the root password. However, the new self-created password can not be used for log in using PuTTY (Windows) or SSH (Linux based OS).

Therefore it is advisable to create the SSH key pair on your system before creating the droplet. And save your public key in "Add your SSH keys" (AYSK) section before hitting the "create" button on "Create Droplets" page.

If you are using PuTTY Key Generator (PKG), copy the public key directly from the PKG's dialogue box, and paste it in "Add your SSH keys" section. Because when PKG saves the public key file, it modifies the preamble and postscript of the public key. Therefore, if you try to use public key from the PKG saved file, AYSK is not able to parse it correctly to be a valid RSA key, and may complain.

greengablesfan33 *December 28, 2017*

0 I have a few questions about the public key and disabling root access.
I use Win SCP quite a lot instead of Nano because as a totally-blind user, I find notepad to be more accessible than using Nano or Win SCP's internal editor. If I disabled root access, how can I change file ownership and permissions, or move and delete files in Win SCP as the root user?
Also, I currently have Windows 10 and Windows Power Shell with Scoop's Open-SSH. I tried using ssh-keygen, but it said it failed to create a directory in a certain path. Luckily, I have a Linux subsystem that is included in the Windows 10 anniversary, and it also comes with the ssh-copy-id , so I was able to use that to copy the publickey to the .ssh folder of my VPS.

Zer00CooL *January 4, 2018*

0 SSH french tutorial : https://www.visionduweb.eu/wiki/index.php?title=SSH

oneLovelifestyles2020 *July 10, 2018*

0 Nice work m8 ! kudos, simple and useful!

SCROLL TO TOP