.

Utility VPS

you can explore and try stuff on you own computer- but why? get you a cheap vps, run an email server and nmap the internet.

- VPS Basics

VPS Basics

Introduction

with a new VPS there are a few configuration steps needed as part of the initial setup. These steps will increase the security and usability of your server, and will give you a solid foundation for subsequent actions.

much of this is stolen from:

https://www.digitalocean.com/community/tutorials/initial-server-setup-with-ubuntu-20-04

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 are not already connected to your server, log in now 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

root is the most powerful account on your *nix system. dont let bad guys steal it from you. give it a strong password and only log in as root in an emergency. These days, there are very few good reasons to be logged in as root. Make a regular user for normal usage, and sudo when you need some magic.

https://xkcd.com/149

Creating a New User

Once you are logged in as **root**, you'll be able to add the new user account. In the future, we'll log in with this new account instead of **root**.

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

```
adduser notroot
```

On a ubuntu system, you will be asked for contact info (name, phone number, etc.) These are optional and probably not needed for a playground vps. Give the account a good password and Enter Key your way through the script. Behind the scenes, Ubuntu is making the new user's home directory and doing some administrative setup.

Now we have a new user account with regular account privileges. Sometimes we need ubuntu to make us sammiches though, so we need to make the new account a **sudoer.** This will allow our normal user to run commands with administrative privileges by putting the word `sudo` before the command.

To add these privileges to our new user, we need to add the user to the **sudo** group. By default, on Ubuntu 20.04, users who are members of 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 username with your new user):

```
usermod -aG sudo notroot
```

Now, when logged in as your regular user, you can type `sudo` before commands to run them with superuser privileges. You will be prompted for your password and the sudo action gets logged in /var/log/auth.log (nice!)

> many VPS systems will drop you right into a root shell after provisioning the server. If you are installing to a cloud vm from ISO, setup will take care of the new user creation/sudo step for you. root user wont even have a password!

Setting Up a Basic Firewall

Use the system firewall to make sure only connections to certain services are allowed. We want to be sure that only allowed applications and services are reachable from the internet. In some cases we want to allow only Certain Hosts to connect.

Your hosting provider might have a firewall available outside of the VM. Make sure you have one activated in any case. I prefer UFW/iptables simply because i can manage it from the vm instead of logging into the cpanel.

on Ubuntu, applications can register their profiles with UFW upon installation. These profiles allow UFW to manage these applications by name. OpenSSH, the service allowing us to connect to our server now, has a profile registered with UFW.

You can see this by typing:

```
ufw app list
```

returning something like:

```
Available applications:
Apache
Apache Full
Apache Secure
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 with:

```
ufw allow OpenSSH
```

Afterwards, enable the firewall:

```
ufw enable
```

UFW will warn you about getting your ssh connection dropped- take a deep breath and...

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

```
ufw status
```

> Status: active

```
To Action From
-- ------ ----
Postfix ALLOW Anywhere
OpenSSH ALLOW Anywhere
Apache ALLOW Anywhere
Apache Secure ALLOW Anywhere
Postfix SMTPS ALLOW Anywhere
Postfix Submission ALLOW Anywhere
Postfix (v6) ALLOW Anywhere (v6)
OpenSSH (v6) ALLOW Anywhere (v6)
Apache (v6) ALLOW Anywhere (v6)
Apache Secure (v6) ALLOW Anywhere (v6)
Postfix SMTPS (v6) ALLOW Anywhere (v6)
Postfix Submission (v6) ALLOW Anywhere (v6)
```

if you install and configure additional services, you will need to adjust the firewall settings to allow traffic in.

https://help.ubuntu.com/community/UFW

https://www.digitalocean.com/community/tutorials/iptables-essentials-common-firewall-rules-and-commands

Now that you have the basics of user access and firewall handled, its time to upgrade your authentication