DigitalOcean - Quick Start Guide

Log in to your account

1. Create Droplet button
2. Name droplet *(isn't the URL)*
    a. droplet01
3. No extra settings
4. Ubuntu 14.04 x32 *(32-bit operating system is recommended for cloud servers with less than 3 GB of RAM)*
    a. Processes consume more memory on 64-bit OS
5. Create Droplet

SSH

- Connect to server securely

PuTTY

- We will create a public key and embed it on our server
- We will have the private key on our own computer that gives us access to the server

http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html

A Windows installer for everything except PuTTYtel
`putty-0.64-installer.exe`

Run installer and keep all defaults (add a Desktop Icon if you want)

Login to your Droplet

1. Start PuTTY
2. Droplet IP address
    a. make sure the port number is 22
3. Enter "Droplet 01" in Saved Sessions field
    a. Save
    b. Open
4. login as: root
5. Password: (`Shift + Insert` to paste password from email)

Create a new user

With root user, it is very easy to mess something up by accident

```
adduser bucky
# enter password > just hit enter to skip over other information
```

- We now have a regular user, but sometimes we do actually want to do admin tasks
- To prevent having to log back into root, we can set up root privileges on our normal account
- Remember, we are still logged in as root

```
#gpasswd is used to administer group, add bucky to sudo group (root privileges)
gpasswd -a bucky sudo
```

He can now use the command: `sudo`

Create a SSH key pair

1. Start PuTTYgen
2. SSH-2 RSA > Generate
3. Save both Public (txt file) and Private key somewhere secure
4. Select key and copy

```
#switch to bucky
su - bucky

#create a new directory called .ssh and restrict its permissions
mkdir .ssh

#change its permissions (owner can read, write and execute)
chmod 700 .ssh

#open this file in text editor
nano .ssh/authorized_keys
```

1. Paste in public key
2. **Ctrl + X** to exit the file
3. **Y** to save the changes that you made
4. **ENTER** to confirm the file name

```
#restrict the permissions of the authorized_keys file (owner can read and write)
chmod 600 .ssh/authorized_keys
```

Close out of PuTTY

Log in as bucky

1. Pageant
2. Add Key > select private key
3. PuTTY
4. Load session > Open
    a. bucky

Change default port

```
sudo nano /etc/ssh/sshd_config
Port 22 -> Port 7777
```

1. **Ctrl + X** to exit the file

2. **Y** to save the changes that you made
3. **ENTER** to confirm

```
#restart the SSH service so that it will use our new configuration
sudo service ssh restart
```

Before you close the session, open a new terminal and test the connection

---

Configure firewall

- Firewall lets you control over connections to your server
- Ubuntu comes with a tool we can use to manage firewall settings called **ufw**
- We will block all traffic except for what we explicitly allow

```
#allow access to 7777 for SSH
sudo ufw allow 7777/tcp

#website
sudo ufw allow 80/tcp

#website using HTTPS
sudo ufw allow 443/tcp

#review changes
sudo ufw show added

#enable the firewall
sudo ufw enable
```

Configure server time zone

```
sudo dpkg-reconfigure tzdata

#use NTP to stay in sync with other servers
sudo apt-get update
sudo apt-get install ntp
```

LAMP - Apache

- Already have Linux
- http://server_ip_address
- You can see we don't have any web server running
- We can use a package manager to install new software really easily: **apt**

```
sudo apt-get update
sudo apt-get install apache2
```

Now go to http://server_ip_address

LAMP - MySQL

```
sudo apt-get install mysql-server php5-mysql

#create its database directory structure where it will store its information
sudo mysql_install_db

#security script to remove common vulnerabilities
sudo mysql_secure_installation
```

- Enter the MySQL root password you just set
- Just hit enter for the rest

LAMP - PHP

```
#install with apt
sudo apt-get install php5 libapache2-mod-php5 php5-mcrypt
```

Right now when a user requests a directory it looks for **index.html**, let's change it to **index.php**

```
#open dir.conf
sudo nano /etc/apache2/mods-enabled/dir.conf
```

Move:

```
DirectoryIndex index.html index.cgi index.pl index.php index.xhtml index.htm
DirectoryIndex index.php index.html index.cgi index.pl index.xhtml index.htm
```

- **Ctrl + X**
- **Y**
- **Enter**

```
#you need to restart the Apache web server for the changes to take effect
sudo service apache2 restart
```

Install PHP Modules

```
#search for additional modules and libraries
apt-cache search php5-

#get more info about a package
apt-cache show php5-json

#to install a package
sudo apt-get install php5-json
```

Create a homepage

```
#homepage
sudo nano /var/www/html/index.php

<?php
echo(“gametime”);
?>
```

- **Ctrl+X > Y > Enter**

```
#info page
sudo nano /var/www/html/info.php

<?php
phpinfo();
?>
```

- **Ctrl+X > Y > Enter**

Now go to http://server_ip_address

```
#if you want to remove the file
sudo rm /var/www/html/info.php
```

Install and Secure phpMyAdmin

```
sudo apt-get install phpmyadmin
```

- apache2
- yes
- Enter database administrators password
- Enter password for phpMyAdmin

```
#enable the php5-mcrypt extension
sudo php5enmod mcrypt

#move the .conf file to the right place
sudo cp /etc/phpmyadmin/apache.conf /etc/apache2/conf-enabled/phpmyadmin.conf

#restart Apache
sudo service apache2 restart
```

- http://104.236.39.49/phpmyadmin/
- root
- password

Secure phpMyAdmin

- phpMyAdmin is a popular target for attackers. We need to secure the application to help prevent unauthorized use
- One of the easiest way of doing this is to place a gateway in front of the entire application
- We can do this using Apache's built-in .htaccess authentication and authorization functionalities

We need to enable the use of .htaccess file overrides by editing our Apache configuration file

sudo nano /etc/apache2/conf-enabled/phpmyadmin.conf

Add this line:

```
<Directory /usr/share/phpmyadmin>
    Options FollowSymLinks
    DirectoryIndex index.php
    AllowOverride All
```

- **Ctrl+X > Y > Enter**

```
#restart Apache
sudo service apache2 restart
```

Create an .htaccess File

Now that we have enabled .htaccess use for our application, we need to create one to actually implement some security. In order for this to be successful, the file must be created within the application directory. We can create the necessary file and open it in our text editor with root privileges by typing:

sudo nano /usr/share/phpmyadmin/.htaccess

And in the file add:

```
AuthType Basic
AuthName "Restricted Files"
AuthUserFile /etc/phpmyadmin/.htpasswd
Require valid-user
```

Create the .htpasswd file for Authentication

Now that we have specified a location for our password file through the use of the AuthUserFile directive within our .htaccess file, we need to create this file. We actually need an additional package to complete this process.

sudo apt-get install apache2-utils

```
#create file and add initial user
sudo htpasswd -c /etc/phpmyadmin/.htpasswd bucky
```

- bucky123

http://104.236.39.49/phpmyadmin/

SFTP

1. FileZilla
2. Edit > Settings
3. SFTP

4. Add Keyfile > Browse to private key
5. OK

then…

1. File > Site manager > New site
    a. *droplet01*
2. Settings
    a. *104.236.39.49*
    b. *7777*
    c. *SFTP*
3. Logon Type: Interactive
4. Connect

Download index.php

To set up a host name: https://www.digitalocean.com/community/tutorials/how-to-set-up-a-host-name-with-digitalocean