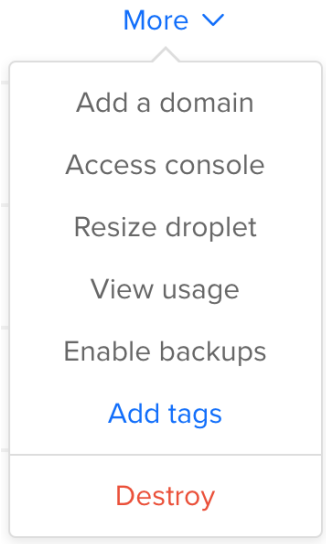
Using Tags on Digital Ocean

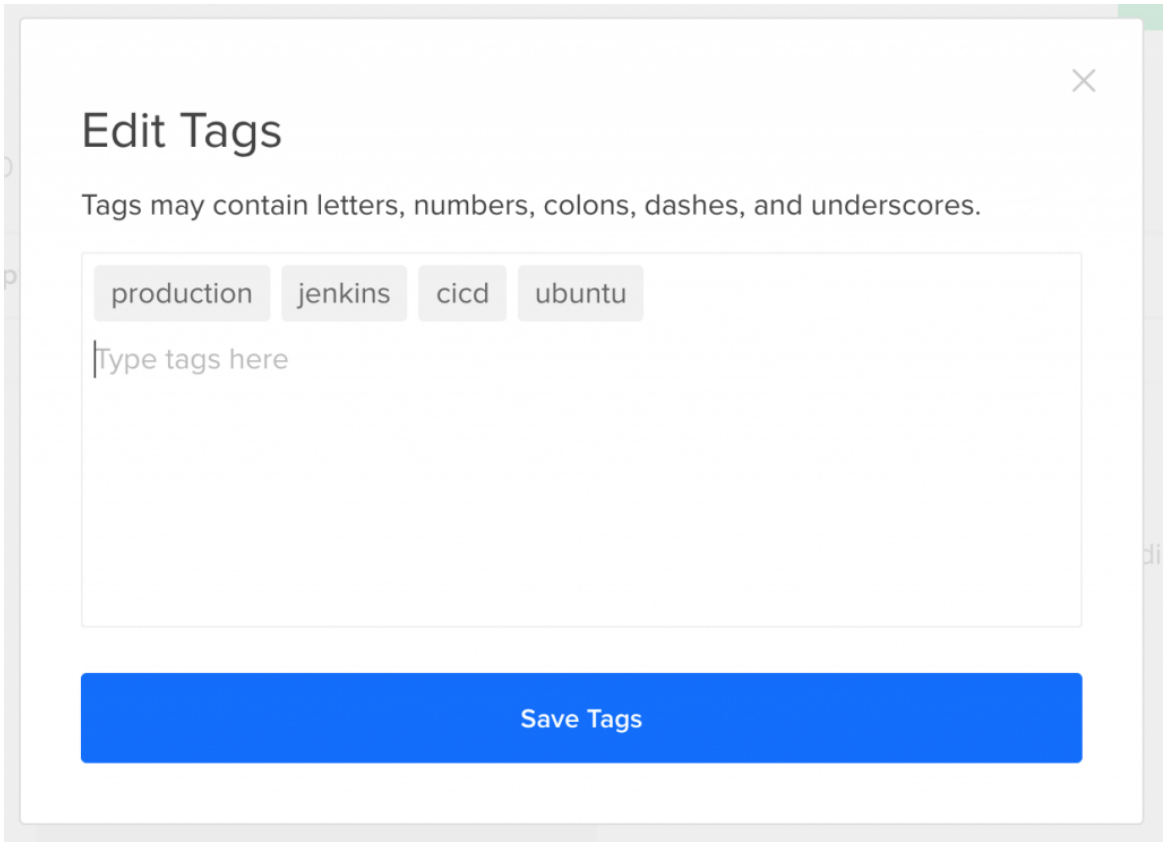
Digital Ocean has recently added the ability to tag your droplets. Tags will let us better organize our workloads with targeted searchable tags to identify features, roles, names, or any type of information at all.

The tags implementation is different on Digital Ocean than some other environments, but you may find it similar to other content platforms rather than the AWS method of assigning tags and values.

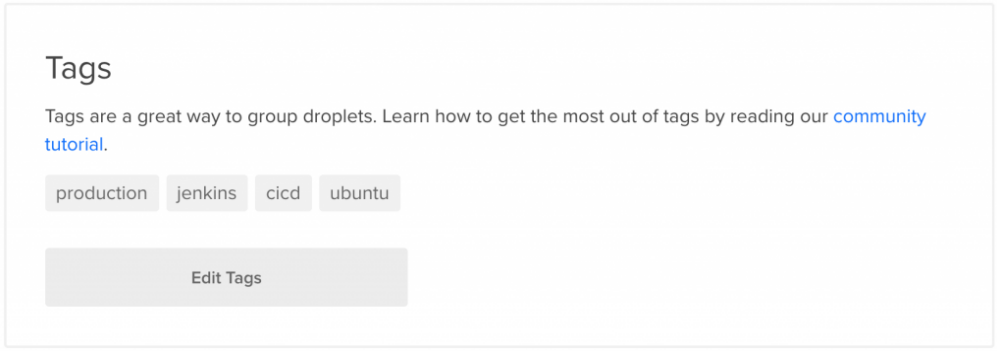
In your Droplets list, click the More menu to get the dropdown. Click on the Add tags links that you see for your droplet:



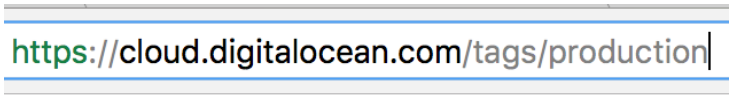
This brings up the Tags editor where you can type some tags which help to identify your droplet. My example shows that I'm assigning the production, jenkins, cicd, and ubuntu tags to this droplet. It happens to be a production Jenkins server:



You'll see the final results once you save the tags, and this is where we can now edit existing tags if needed:



Searching for tags is a little different than you may think. The search field on the droplets page is still tied to the name of the droplet. To use tags for categorization and search, you can use the tags URL here:



This gives us everything tagged as production which you can see by the tags column:

By clicking on the +3 link, you will also see the additional tags assigned to that droplet:

All of the links and tags are clickable, which brings you to the categories that you'll want to see. By crafting your URL, you can choose which ones to query on.

That's the basics of tagging for Digital Ocean, and hopefully you'll find value in this. More and more of the work that we do in cloud and data center platforms can make use of tags for numerous administrative functions. This is the time to start getting used to using them.

Deploying a Jenkins server on Digital Ocean with Vagrant

One of the most popular Continuous Integration (CI) tools out there today is a nifty product named Jenkins. For those who want to take a look at how to utilize Jenkins, but you may not have had a place to try it out, I have just what you need!

You'll need a few things for the build:

1. Vagrant – http://www.vagrantup.com/downloads.html (Free)
2. Vagrant DigitalOcean plugin (https://github.com/smdahlen/vagrant-digitalocean)
3. A DigitalOcean account – If you don't have one yet, go here: http://discopos.se/GetDigitalOcean
4. Your DigitalOcean API Keys (both Client and Secret) and SSH key
5. Git – you'll have to check out my Github repo to deploy the system

Let’s get started and show you get all the information you need and how to put these components together!

NOTE: This will create a server that will be a 1GB DigitalOcean Droplet which costs 10$ per month if left active continuously

Getting the GitHub repository

Depending on your system, you’ll need to either pull down the code via the GitHub GUI, or the command line. The repository is easy to grab in the command line like so:

```
git clone https://github.com/discoposse/Vagrant-DigitalOcean-Jenkins.git
```

Now that you have that sorted out, you will need to head out to the command line and begin building our Vagrant configuration. Although we have the files needed to deploy our server, there is some

Getting your DigitalOcean Key info

You will have to log in to http://digitalocean.com and under your account page, you will find the API option in the left hand menu which shows your API key information. You will see the **Client ID** and your **API Key**.

In the event that you see your API Key as hidden (shown below) you will have to use the **Generate New Key** button at the top right. NOTE: Each time you regenerate an API key, it will invalidate your previous key so you should keep this key somewhere safe as it gives full access to your account.

Now that you have your API Key and Client ID, you also need your matching SSH key that was used during the setup of your account. For all the SSH key information, you can use this guide at DigitalOcean: https://www.digitalocean.com/community/articles/how-to-use-ssh-keys-with-digitalocean-droplets

Setting up your Vagrantfile

In the folder where you download there is a file named **Vagrantfile.sample** that you can copy, paste and rename to **Vagrantfile** with no file extension.

This is the folder where you will be storing your **id_rsa** and **id_rsa.pub**. Once you have those to SSH files there and your newly copied **Vagrantfile** the folder should look like this:

Inside your **Vagrantfile** you will be putting in your customer specific information which are your Client Key and your API Key. Just replace the text inside the **Vagrantfile** in the appropriate lines:

That’s all the preparation we need for our file. Now let’s make sure our Vagrant configuration is

ready to deploy to DigitalOcean.

Installing the DigitalOcean Vagrant plugin

In your command prompt or shell, change directory to the folder where your GitHub repository was cloned. From there, you can also deploy (or confirm the deployment) of the Vagrant plugin for DigitalOcean (https://github.com/smdahlen/vagrant-digitalocean).

In the shell, run this command:

```
vagrant plugin install vagrant-digitalocean
```

Now we are ready to launch our DigitalOcean Droplet which will run our Jenkins server.

Deploying Jenkins with Vagrant and the DiscoPosse Quick Build

In the command shell, you have one simple command to run which is as follows:

```
vagrant up –provider=digital_ocean
```

This will trigger the deployment to your DigitalOcean account. The script will begin as shown here:

The script will run for about 5-10 minutes as it updates the instance with the Ubuntu updates and Jenkins server code. Once it is completed you will see a message similar to this indicating that your droplet is ready.

Now we will launch our browser to view the Jenkins application interface. First you will need the IP address of the instance you have launched.

To get the IP address, you can view your DigitalOcean control panel and under the Droplets menu you will see the instance information like the sample here:

Open a browser window using the the IP address with port 8080 to access Jenkins:

```
http://yourIPaddress:8080
```

The first thing that you will want to do is to add users and authentication requirements, so I recommend that you click the Manage Jenkins link to get that started:

Under the Jenkins management screen you can enable security in a number of ways. This will be up to you as to how much you choose to lock down your application. Just click the **Setup Security** button to do this:

And just like that you have your first Jenkins server on DigitalOcean. It is just that easy thanks to Vagrant!

Getting rid of your Jenkins instance

Once you get a chance to test things out, or if you want to rebuild your machine, you can easily remove the server using Vagrant with this command:

```
vagrant destroy -f
```

This makes it very simple for you to create and destroy your lab environment as needed. Especially if you are like me and have a number of projects on the go that don’t always get full attention. By using vagrant destroy, you are no longer being charged for your instance.

I hope that you find this helpful and that it can help you on your journey to discover Jenkins.