The Day Your Laptop Became A Micro-Datacenter

Did you know that you can push your laptop from being an ordinary computer to a Micro-Datacenter powering a Kubernetes cluster and then back to an ordinary computer in only 15 minutes?

Yes, you read right.

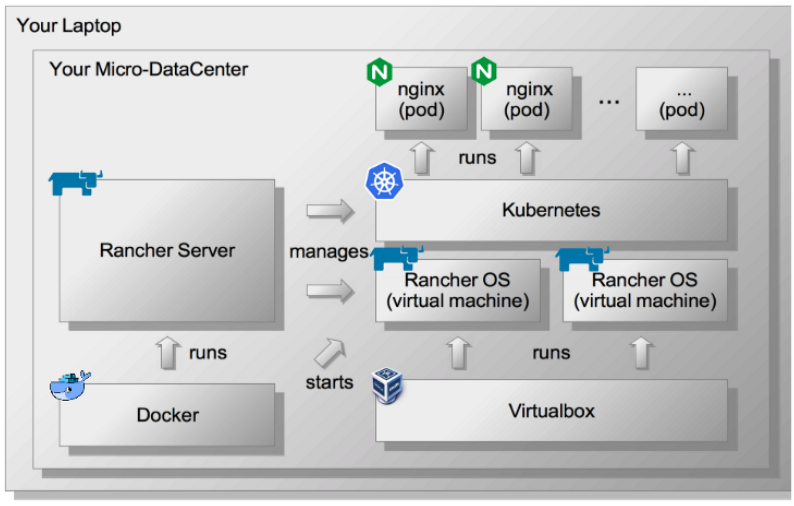
15 minutes to get your own private Kubernetes cluster up and running on your laptop.

Microservices and Kubernetes are the hype.

But how do you get started with learning more about these two exciting topics?

I will show you every step of the way and get you up to speed with a learning environment on your laptop. By the end of this post you will be all set to experiment and explore Kubernetes on your laptop.

The diagram below gives you an overview of what your Micro-Datacenter will look like when we are done setting it up.



We will leverage a virtualisation layer (consisting of Docker containers and virtual machines) running on your laptop hardware.

On this virtualisation layer we will run Rancher which will help us with setting up and managing our Micro-Datacenter as well as the Kubernetes cluster.

Here are the 5 short steps to build your Micro-Datacenter

1. Install the dependencies (Docker & Virtualbox)
2. Install Rancher Server
3. Create a Kubernetes environment in Rancher
4. Add two nodes (virtual machines) to the Kubernetes cluster
5. Run a 'Hello World' nginx service in Kubernetes

First things first.

1. Install the Dependencies

There are only two dependencies to get us on our way:

1. We need Docker installed on your laptop (click here and follow the instructions to install Docker)
2. We need Virtualbox installed on your laptop (click here and follow the instructions to install Virtualbox)

Take a minute and install them both if you have not done so already.

When you are done with installing go ahead and start Docker.

Docker is running

The screen shots and terminal commands in this post are taken from a Mac. However, with very few exceptions (which I will point out) you should be fine following along on either Linux or Windows.

Let's get started (check your watch :)

2. Install Rancher Server

Rancher Server will manage our Micro-Datacenter and our Kubernetes cluster. Let's fire up Rancher in a Docker container.

Open a terminal on your laptop and copy-paste the following command (one line):

```
> docker run -d --restart=unless-stopped -p 8080:8080
rancher/server:v1.6.3
```

```
### The Rancher Server will take some time to get up and
running.
### You can tail the logs to check Rancher's progress
> docker ps
CONTAINER ID        IMAGE
4a3dcbaaa4ba        rancher/server:v1.6.3
```

```
> docker logs -f 4a3dcbaaa4ba
```

Once Rancher is up and running you can point your browser to localhost port 8080 and access the Rancher Server's web interface.

Later on we want our two Kubernetes nodes (virtual machines) to connect to the Rancher Server. The two nodes will not be able to use the localhost address to connect. Therefore we need to add an alias to the localhost interface on our laptop. Let's assign a private IP address as an alias.

I have chosen **10.145.10.1** as a random private IP address. Note it does not matter what private IP address you choose. You will be fine as long as you stay clear of the more common ones (like 192.168.... or 10.0....). Staying clear of these ranges of private IPs will help avoiding networking conflicts.

Note for Windows users: Assigning the alias might be somewhat different on Windows. Please use caution and read up on the details for your OS. For Linux users the below commands should look very similar.

```
### Add the alias

> sudo ifconfig lo0 alias 10.145.10.1/24

### Check with ifconfig to see whether the alias has been
added

> ifconfig
lo0: flags=8049<UP,LOOPBACK,RUNNING,MULTICAST> mtu 16384
 options=1203<RXCSUM,TXCSUM,TXSTATUS,SW_TIMESTAMP>
 inet 127.0.0.1 netmask 0xff000000
 inet6 ::1 prefixlen 128
 inet6 fe80::1%lo0 prefixlen 64 scopeid 0x1
 inet 10.145.10.1 netmask 0xffffff00
 nd6 options=201<PERFORMNUD,DAD>
```

Verify that you can reach the Rancher Server web interface by pointing your browser to the alias on 10.145.10.1 port 8080

3. Create the Kubernetes environment in Rancher

At the top left corner on the Rancher web interface click on *Environment* then select *Manage Environments* > *Add Environment*

Give the new environment a name (ie 'Kubernetes Demo'). Choose the *Kubernetes* Environment Template, then scroll down and click 'Create'.

Create a Kubernetes Environment

You will now see new *'Kubernetes Demo'* environment in the *Environment* drop down in the top left corner. Click on it.

At this stage Rancher is waiting for us to add infrastructure nodes to the newly created Kubernetes environment.

Rancher waits for infrastructure nodes to be added

4. Add two nodes (virtual machines) to the Kubernetes cluster

We create the two virtual machines using the *docker-machine* command from Docker (note you could add as many nodes as your hardware can support, but for the demo two are enough).

For our two Kubernetes nodes we could use any Linux distro machine image that comes with Docker installed. However, Rancher distributes a tiny container optimised image called Rancher OS. We'll use Rancher OS. It is perfect for our use case.

Go back to your terminal and copy-paste the command from the box below (one line).

Note: **run the command twice** once with **kube-box-1** and then with **kube-box-2** to create our two nodes.

```
### Create a Rancher OS virtual machine with 3G of RAM (one
line)
```

```
> docker-machine create -d virtualbox --virtualbox-memory
"3072" --virtualbox-boot2docker-url
https://releases.rancher.com/os/latest/rancheros.iso kube-
box-1


### Run the command again now for the second node

> docker-machine create -d virtualbox --virtualbox-memory
"3072" --virtualbox-boot2docker-url
https://releases.rancher.com/os/latest/rancheros.iso kube-
box-2
```

Rancher OS comes with Docker pre-installed. Unfortunately, the default version of Docker is newer than Kubernetes likes its Docker to be.

We have to switch Docker on the two nodes to a suitable version (1.12.6).

Login to each of the two nodes and change the Docker version.

```
### ssh to kube-box-1

> docker-machine ssh kube-box-1

### and switch the Docker version to 1.12.6 then logout

[docker@kube-box-1 ~]$ sudo ros engine switch docker-1.12.6
[docker@kube-box-1 ~]$ exit

### then do the same for the second node

> docker-machine ssh kube-box-2

[docker@kube-box-2 ~]$ sudo ros engine switch docker-1.12.6
[docker@kube-box-2 ~]$ exit
```

Fantastic. Our two nodes are now ready to join the Kubernetes cluster.

In the next step we will run a little script on both of the two nodes to register them with Rancher and join the Kubernetes environment.

For the registration script we will need the nodes' IP addresses (note your nodes' IP addresses might be different).

```
### Retrieve the IP address for kube-box-1

> docker-machine ip kube-box-1
192.168.99.100

### and the IP address for kube-box-2

> docker-machine ip kube-box-2
192.168.99.101
```

We have got all the info we need. It's time to go back to the Rancher web interface and click on **'Add a host'**

Add a host

Keep the default '*Custom*' setting for the new node.

Then scroll down to step 4. There specify the IP address for **kube-box-1** as retrieved above (**192.168.99.100** or whatever the value is that *docker-machine* has allocated in your case).

Rancher needs the IP address to communicate with its infrastructure nodes.

Enter the node's IP address and copy the registration script

Copy the command from step 5. This is the registration script to run the Rancher agent on **kube-box-1**.

Go back to the terminal on your laptop and login to **kube-box-1** using the *docker-machine ssh* command we already used above.

```
### ssh to kube-box-1
```

```
> docker-machine ssh kube-box-1
```

```
### on kube-box-1 run the script from the Rancher web
interface
```

```
[docker@kube-box-1 ~]$ sudo docker run -e
CATTLE_AGENT_IP="192.168.99.100"  -d --privileged -v
/var/run/docker.sock:/var/run/docker.sock -v
/var/lib/rancher:/var/lib/rancher rancher/agent:v1.2.1
http://10.145.10.1:8080/v1/scripts/F6FC06D291DDADE7F0B7:1483
142400000:8dr5IQ1NsrNG2pEX2hLh7XPppI
```

```
### wait for the command to complete then logout
```

```
[docker@kube-box-1 ~]$ exit
```

Do all of the above (starting from 'Add a host') again to register **kube-box-2.**

Important! Remember to copy-paste the IP address for **kube-box-2** (ie **192.168.99.101**) in step 5. Bad things will happen if you register kube-box-2 with kube-box-1's IP address.

Rancher is now busy provisioning the two nodes with the Kubernetes services.

Switch to *Infrastructure > Hosts* to watch the Kubernetes services coming up on the two nodes.

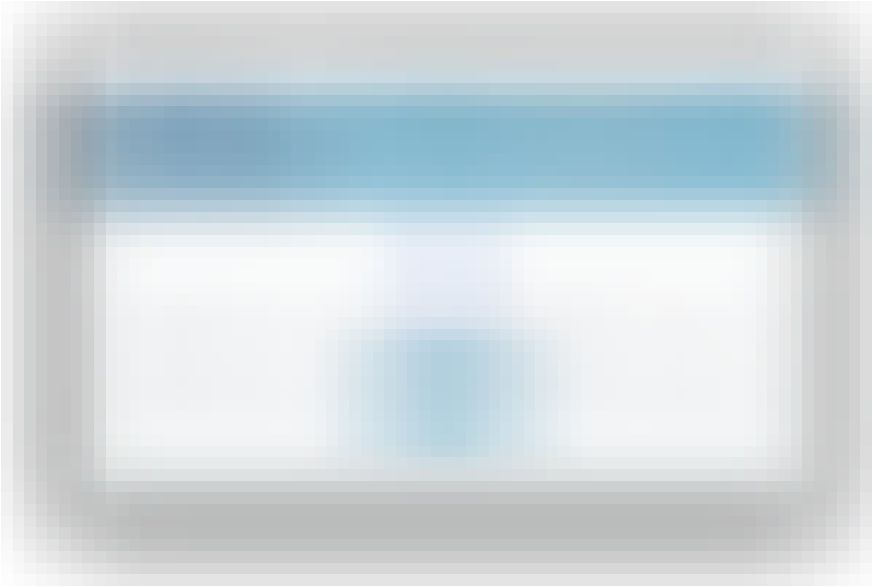
Rancher provisions Kubernetes services on the two nodes

It will take some time for Rancher to download and install the images for the Kubernetes services on the two nodes.

Be patient. Depending on your internet connection this might take quite some time. There are several ways to tell when your Kubernetes cluster is configured and up and running.

One way is when Rancher will start providing a link to open the Kubernetes dashboard (*Kubernetes > Dashboard*).

You have some time while Rancher is hard at work spinning up your Kubernetes cluster. Have a play with Rancher's web interface to see what else it offers that we did not discuss yet.



Kubernetes is ready. You might want to checkout the Kubernetes dashboard.

5. Run a 'Hello World' nginx service in Kubernetes

Now, to the exiting part! We have Kubernetes running in our Micro-Datacenter.

In good old fashioned tradition let's run the 'Hello World' of Microservices: A simple nginx web server (serving a default web page).

Kubernetes provides the *kubectl* command line tool to configure its resources.

With Rancher Server we can either choose to

- run *kubectl* from the console in the web interface or
- if you have *kubectl* installed on your laptop then Rancher gives us the *kubectl* configuration file to copy over to the laptop.

Both options are available from the *Kubernetes* > *CLI* menu in the Rancher web interface.

For simplicity we will use Rancher's web console for running *kubectl*.

Of course you are free to install kubectl on your laptop and configure it with the config provided by Rancher (click on the *'Generate Config'*

button as shown on the screen shot below). Having *kubectl* installed on your laptop will come in handy when you start doing more interesting stuff with your Kubernetes cluster later.

kubectl shell in the Rancher web interface (or check out the config file if you run kubectl on your laptop)

For your convenience I created a github repo with the Kubernets configuration files (**.yaml*) to run the nginx 'Hello World' example.

All you need to run the nginx example (a Kubernetes deployment and Kubernetes service) is executing the two '*kubectl create -f …*' commands in Rancher's web console.

Clone the github repo if you either intend to run *kubectl* from your laptop or you want to study the content of the two *yaml* files.

```
### Copy-paste the commands and enter them on Rancher's web
console

### Create a deployment (spinning up two nginx pods)

> kubectl create -f
https://raw.githubusercontent.com/danaschwanden/micro-
datacenter/master/HelloWorld/nginx-deployment.yaml

deployment "dep-nginx" created


### Check to see the deployment we just created
```

```
> kubectl get deployments

NAME      DESIRED CURRENT UP-TO-DATE AVAILABLE AGE
dep-nginx 2       2       2          2         6s

### Check to see the two pods the deployment created

> kubectl get pods

NAME                         READY STATUS  RESTARTS AGE
dep-nginx-3655994670-2gks0 1/1   Running 0        10s
dep-nginx-3655994670-53dfd 1/1   Running 0        10s

### Create a service to expose the nginx services on the two
nodes

> kubectl create -f
https://raw.githubusercontent.com/danaschwanden/micro-
datacenter/master/HelloWorld/nginx-service.yaml

service "svc-nginx" created


### The service is configured to use "NodePort"
### Kubernetes master will allocate a random port from a
### flag-configured range (default: 30000-32767)
### Let's check on what port the service exposes nginx

> kubectl get services

NAME      CLUSTER-IP    EXTERNAL-IP PORT(S)        AGE
svc-nginx 10.43.164.192 <nodes>     8080:32767/TCP 12s
```

Note the output of the *'kubectl get services'* command above. It tells us two things:

- The nginx service (*svc-nginx*) is exposed on the external IP ***<nodes>***
- on port ***32767*** (might be different in your case).

This means nginx is now available on both our ***nodes*** on port ***32767***.

Let's verify that we can reach nginx by pointing the browser to either of the nodes' IP addresses at the port the Kubernetes service has assigned.

For example **http://192.168.99.100:32767** to access nginx via **kube-box-1**. Note that your IP address and port might be different. Change them accordingly.

Et voila

Congratulations. Your laptop is running a Micro-Datacenter with Kubernetes and nginx as a simple show case service.

Over to you now to play with the setup at your heart's content. We barely touched the surface of what can be done. Go wild.

I hope you will enjoy your new datacenter powers.

Once you are done you probably want to clean up and free up the resources. Cleaning up is just as easy as it was to set up the environment. Simply follow the 5 steps below.

Cleaning up

As promised we want to leave your laptop as pristine as we found it at the beginning of this post.

Let's get the broom out.

a) Stop the Kubernetes service and deployment.

```
### In the Rancher web interface console run

> kubectl delete service svc-nginx
service "svc-nginx" deleted

> kubectl delete deployment dep-nginx
deployment "dep-nginx" deleted
```

b) Stop the two nodes and remove them

```
### In the terminal on your laptop run

> docker-machine stop kube-box-1

> docker-machine stop kube-box-2

> docker-machine rm kube-box-1

> docker-machine rm kube-box-2
```

c) Remove the alias on your laptop's localhost interface. *If you are on Windows or Linux this step might be slightly different. Please use caution and read up on it for your OS.*

```
### In the terminal on your laptop run

> sudo ifconfig lo0 -alias 10.145.10.1
```

d) Stop the Rancher Server container and remove it

```
### In the terminal on your laptop run

> docker ps
CONTAINER ID IMAGE
4a3dcbaaa4ba rancher/server:v1.6.3

> docker stop 4a3dcbaaa4ba
> docker rm 4a3dcbaaa4ba

> docker images
REPOSITORY         TAG             IMAGE ID
rancher/server     v1.6.3          b4c15b304585

> docker rmi b4c15b304585
```

Done. I hope you enjoyed the journey (have you checked the time?)

Useful Links

- The Rancher post that inspired this one
- The Rancher documentation
- The Rancher OS documentation
- The Kubernetes documentation
- The nginx documentation