

Using Docker Machine

docker-machine is a utility that can automate the installation of Docker on a remote host. You can use docker-machine to easily deploy Docker to a virtual machine in the Rapid Access Cloud.

- [Setting up Machine on Your Workstation](#)
- [Create a Security Group](#)
- [Determine the flavor and image you want to use](#)
- [Provisioning Docker host](#)
- [Configuring and building the application image.](#)
- [Deploying containers to Docker host remotely](#)

Setting up Machine on Your Workstation

1. Download and Install a compatible version of docker from <https://www.docker.com/>
2. Install the [OpenStack command-line tools](#).
3. Download an [openrc file](#).
 - a. Set the **OS_TENANT_ID** and **OS_TENANT_NAME** to the same values as **OS_PROJECT_ID** and **OS_PROJECT_NAME**
 - b. Add **export OS_DOMAIN_ID=default**

Create a Security Group

Log in to the RAC Dashboard. Then, either create a new security group or add new values to the default security group to allow port 80 and port 2376:

Displaying 4 items

☐	Direction	Ether Type	IP Protocol	Port Range	Remote IP Prefix	Remote Security Group	Actions
☐	Egress	IPv6	Any	Any	::/0	-	Delete Rule
☐	Egress	IPv4	Any	Any	0.0.0.0/0	-	Delete Rule
☐	Ingress	IPv4	TCP	80 (HTTP)	101.102.103.104/32	-	Delete Rule
☐	Ingress	IPv4	TCP	2376	101.102.103.104/32	-	Delete Rule

Displaying 4 items



Port 80 used by HTTP for our app

Port 2376 is used by docker application

Determine the flavor and image you want to use

In a terminal, use the below commands to see the available flavors and images in RAC.

```
$ source /path/to/your/rc/file
$ openstack flavor list
$ openstack image list
```

For the purpose of this tutorial, we will use an `m1.small` flavor and the `Ubuntu 18.04` image.

Provisioning Docker host

Run the following command:

```
$ docker-machine create -d openstack --openstack-flavor-name m1.small --openstack-image-name "Ubuntu 18.04" --
openstack-floatingip-pool public --openstack-sec-groups default --openstack-ssh-user ubuntu RAC
```

Once the machine is successfully created run below command to see if Instance is running.

```
$ docker-machine ls
```

And you should see output similar to the following:

NAME	ACTIVE	DRIVER	STATE	URL	SWARM	DOCKER	ERRORS
RAC	-	openstack	Running	tcp://162.246.156.100:2376		v20.10.8	

After the host is provisioned , check the server environment with the following command:

```
$ docker-machine env RAC
```

```
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://162.246.156.100:2376"
export DOCKER_CERT_PATH="/Users/omgarcia/.docker/machine/machines/RAC"
export DOCKER_MACHINE_NAME="RAC"
# Run this command to configure your shell:
# eval $(docker-machine env RAC)
```

Then run:

```
$ eval $(docker-machine env RAC)
```

Configuring and building the application image.

Run the below commands to create an application folder:

```
$ mkdir myapp
$ cd myapp
```

Create a file named Dockerfile:

```
$ vi Dockerfile

# Alternatively, you can use nano:
$ nano Dockerfile
```

Copy below script into your Dockerfile:

```
FROM ubuntu:16.04
MAINTAINER yourname <youremail>
RUN apt-get update && apt-get install -y apache2 && apt-get clean && rm -rf /var/lib/apt/lists/*
ENV APACHE_RUN_USER www-data
ENV APACHE_RUN_GROUP www-data
ENV APACHE_LOG_DIR /var/log/apache2
ENV APACHE_PID_FILE /var/run/apache2/apache2.pid
ENV APACHE_RUN_DIR /var/run/apache2
ENV APACHE_LOCK_DIR /var/lock/apache2
ENV APACHE_LOG_DIR /var/log/apache2
RUN mkdir -p $APACHE_RUN_DIR
RUN mkdir -p $APACHE_LOCK_DIR
RUN mkdir -p $APACHE_LOG_DIR
COPY index.html /var/www/html
EXPOSE 80
CMD ["/usr/sbin/apache2", "-D", "FOREGROUND"]
```

In the same application folder, create a file called `index.html`:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <title>First APP</title>
</head>
<body>
  <h1>This is an imageapp container</h1>
</body>
</html>
```

Next, run:

```
$ docker build -t appimage .
```

Verify the image exists:

```
$ docker images
```

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
appimage	latest	20921033f808	9 seconds ago	234MB
ubuntu	16.04	b6f507652425	3 days ago	135MB

Deploying containers to Docker host remotely

To deploy the application image to your Docker host, run:

```
$ docker run --name myapp -i -t -p 80:80 appimage
```

Verify the container is running with the following command:

```
$ docker ps -a
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
b3b6f066d5ab	appimage	"/usr/sbin/apache2 -..."	58 minutes ago	Up 58 minutes	0.0.0.0:80->80/tcp, :::80->80/tcp	myapp

Type in the IP address of your instance (either a Floating IP address or your IPv6 address) in your web browser to see your container running:



This is an imageapp container

You can also log in to your docker host directly by running:

```
$ docker-machine ssh RAC
```