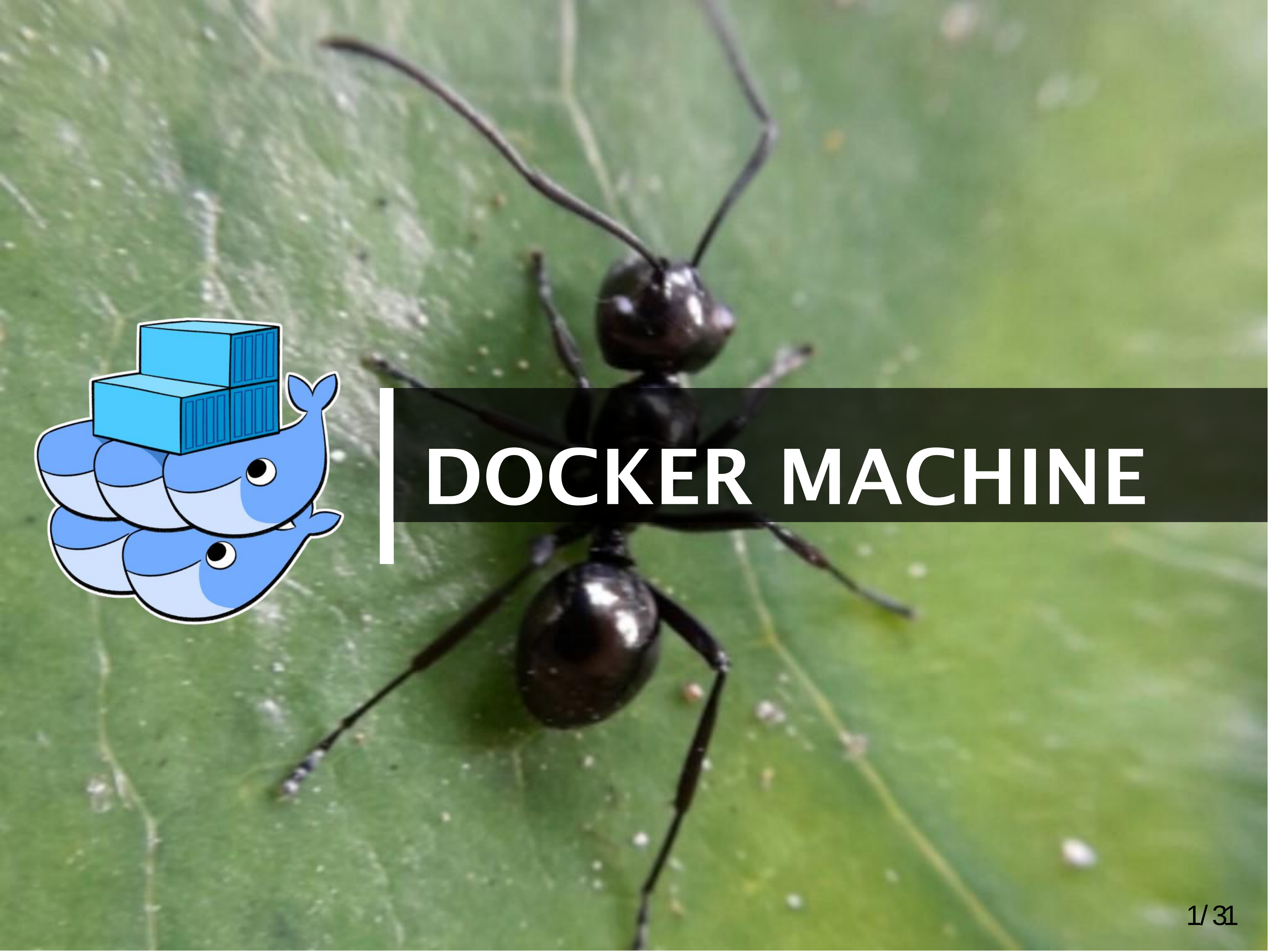


DOCKER MACHINE

Docker Machine vs. Docker Engine

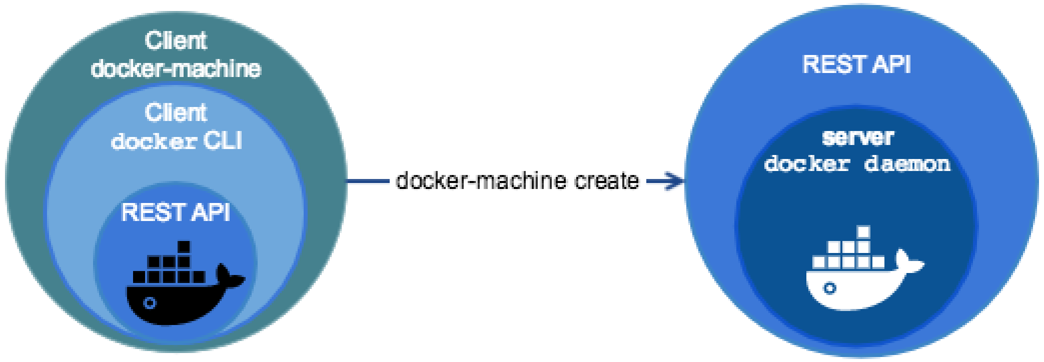
Docker Machine is a tool for provisioning and managing your Dockerized hosts (hosts with Docker Engine on them). Typically, you install Docker Machine on your local system. Docker Machine has its own command line client docker-machine and the Docker Engine client, docker.

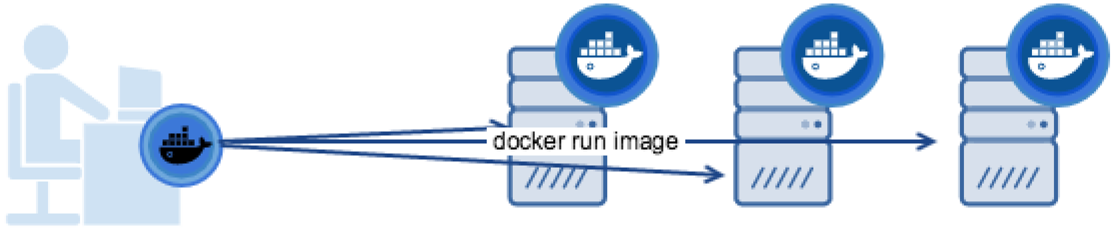
You can use Machine to install Docker Engine on one or more virtual systems. These virtual systems can be local (as when you use Machine to install and run Docker Engine in VirtualBox on Mac or Windows) or remote (as when you use Machine to provision Dockerized hosts on cloud providers).4

The Dockerized hosts themselves can be thought of, and are sometimes referred to as, managed "machines".
Using docker-machine commands, you can start, inspect, stop, and restart a managed host, upgrade the Docker client and daemon, and configure a Docker client to talk to your host.

Point the Machine CLI at a running, managed host, and you can run docker commands directly on that host. For example, run docker-machine env default to point to a host called default, follow on-screen instructions to complete env setup, and run docker ps, docker run hello-world, and so forth.

Docker Machine





Lets Try

Install

```
$ curl -L https://github.com/docker/machine/releases/download/v0.6.0/docker-machine-
$ chmod +x /usr/local/bin/docker-machine

$ docker-machine version
docker-machine version 0.7.0, build a650a40

$ docker-machine ls
NAME    ACTIVE    DRIVER    STATE    URL    SWARM    DOCKER    ERRORS
```

Create Machine

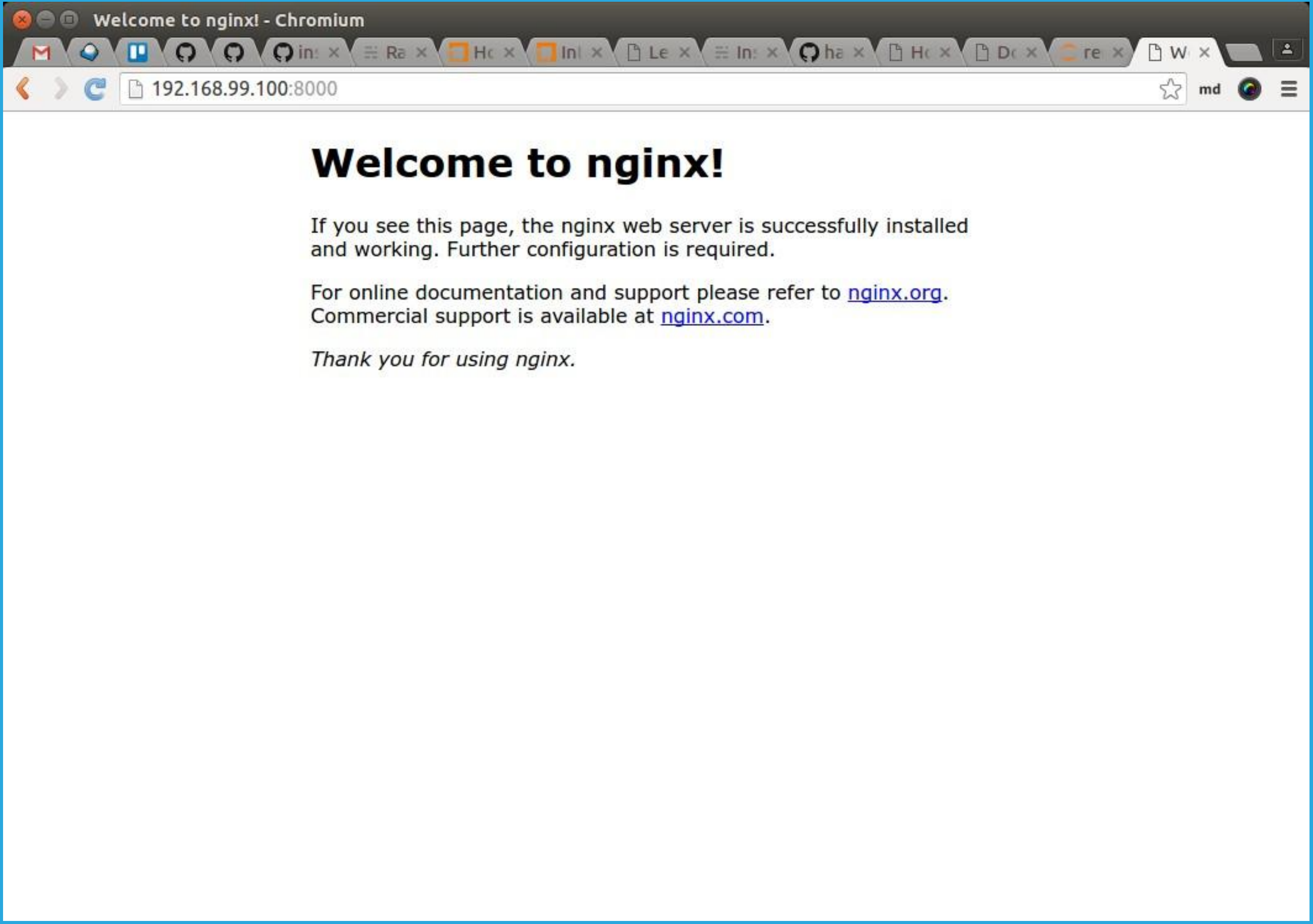
```
$ docker-machine create --driver virtualbox default
Creating CA: /home/em/.docker/machine/certs/ca.pem
Creating client certificate: /home/em/.docker/machine/certs/cert.pem
Running pre-create checks...
(default) Image cache directory does not exist, creating it at /home/em/.docker/mach
(default) No default Boot2Docker ISO found locally, downloading the latest release..
(default) Latest release for github.com/boot2docker/boot2docker is v1.11.1
(default) Downloading /home/em/.docker/machine/cache/boot2docker.iso from https://gi
(default) 0%....10%....20%....30%....40%....50%....60%....70%....80%....90%....100%
Creating machine...
(default) Copying /home/em/.docker/machine/cache/boot2docker.iso to /home/em/.docker
(default) Creating VirtualBox VM... (default) Creating SSH key... (default) Starting
(default) Check network to re-create if needed...
(default) Found a new host-only adapter: "vboxnet1"
(default) Waiting for an IP...
Waiting for machine to be running, this may take a few minutes...
Detecting operating system of created instance...
Waiting for SSH to be available...
Detecting the provisioner...
Provisioning with boot2docker...
Copying certs to the local machine directory...
Copying certs to the remote machine...
Setting Docker configuration on the remote daemon...
Checking connection to Docker...
Docker is up and running!
To see how to connect your Docker Client to the Docker Engine running on this virtua

$ docker-machine  ls
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM   DOCKER
default   -        virtualbox   Running   tcp://192.168.99.100:2376           v1.11.
```

Connect

```
$ docker-machine env default
export DOCKER_TLS_VERIFY="1"
export DOCKER_HOST="tcp://192.168.99.100:2376"
export DOCKER_CERT_PATH="/home/em/.docker/machine/machines/default"
export DOCKER_MACHINE_NAME="default"
# Run this command to configure your shell:
# eval $(docker-machine env default)

$ eval $(docker-machine env default)
$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE

$ env|grep DOCKER
DOCKER_HOST=tcp://192.168.99.100:2376
DOCKER_MACHINE_NAME=default
DOCKER_TLS_VERIFY=1
DOCKER_CERT_PATH=/home/em/.docker/machine/machines/default

$ docker-machine  ls
NAME      ACTIVE   DRIVER       STATE     URL                         SWARM   DOCKER
default   *        virtualbox   Running   tcp://192.168.99.100:2376           v1.11.
```

Work

```
$ docker run busybox echo hello world
Unable to find image 'busybox:latest' locally
latest: Pulling from library/busybox ... 385e281300cc: Pull complete ... a3ed95caeb0
Digest: sha256:4a887a2326ec9e0fa90cce7b4764b0e627b5d6afcb81a3f73c85dc29cea00048
Status: Downloaded newer image for busybox:latest
hello world

$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
busybox             latest              47bcc53f74dc        6 weeks ago         1.11

$ docker-machine  ip  default
192.168.99.100

$ docker ps -a
CONTAINER ID        IMAGE               COMMAND             CREATED             STA
0fb8afca05ea        busybox             "echo hello world"  2 minutes ago       Exi

$ docker run -d -p 8000:80  nginx
Unable to find image 'nginx:latest' locally
latest: Pulling from library/nginx
efd26ecc9548: Pull complete ... 8ddc2d7beb91: Pull complete
Digest: sha256:2ca2638e55319b7bc0c7d028209ea69b1368e95b01383e66dfe7e4f43780926d
Status: Downloaded newer image for nginx:latest
a8eb8f257cc1a747ff5bd30c056fcf5fe173de8fb0781265eed595ab7cd69b40

$ docker ps
CONTAINER ID        IMAGE               COMMAND                 CREATED
a8eb8f257cc1        nginx               "nginx -g 'daemon off"  10 seconds ago
```



Welcome to nginx!

If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

Thank you for using nginx.

Work

```
# or
$ curl $(docker-machine ip default):8000

$ docker-machine stop
Stopping "default"...
Machine "default" was stopped.
# without explicit name -> default

$ env | grep DOCKER
DOCKER_HOST=tcp://192.168.99.100:2376
DOCKER_MACHINE_NAME=default
DOCKER_TLS_VERIFY=1
DOCKER_CERT_PATH=/home/em/.docker/machine/machines/default

$ docker images
An error occurred trying to connect: Get https://192.168.99.100:2376/v1.23/images/js

$ eval $(docker-machine env -u)
$ env | grep DOCKER
$ docker images
REPOSITORY              TAG              IMAGE ID          CREATED
composetest_web         latest           e7d62ba30c20      4 days ago
web                     latest           d6f25a9bf632      4 days ago
redis                   latest           9a450ae418d8      4 days ago
```

Todays agenda

- What is docker-machine?
- Which one to use? docker-machine vs docker remote?
- Quiz

Cool, can deploy it on many hosts the same way. But..

- manage through ssh on each server
- if new server comes up, install os, configure stuff, install and manage dockers
- if it is cloud service, manage vendor specific cloud service since they are heterogeneous
- CRUD operations on VMs requires the development of SDK or vendor specific managements

So what does it do? ...in short

- maps local docker cmd to docker command on remote machine
- execute the same commands you could do locally on remote machine
- execute CRUD operations on VMs
- switch between hosts: physical, virtual or cloud seamlessly

Prerequisites

1. VirtualBox
2. docker-engine
3. access to internet

Install docker-machine on Linux

```
curl -L
https://github.com/docker/machine/releases/
do  wnload/v0.3.0/docker-machine_linux-
amd64 >
/usr/local/bin/docker-machine
```

Install docker-machine on MacOS

```
curl -L
https://github.com/docker/machine/releases/
do  wnload/v0.3.0/docker-machine_darwin-
amd64
> /usr/local/bin/docker-machine
```

Install docker-machine on Windows

Go here and download version for your platform:
https://docs.docker.com/machine/

Hello docker-machine

```
docker-machine create --driver virtualbox
dev1  docker-machine ls
eval "$(docker-machine env
dev1)"  docker run busybox echo
hello world
```

Hello docker-machine again

docker-machine create --driver virtualbox dev2 docker-machine ls
eval "$(docker-machine env dev2)" docker run busybox echo hello world

So what we have?

If you do “docker-machine ps” you will have to different machines with one containers inside.

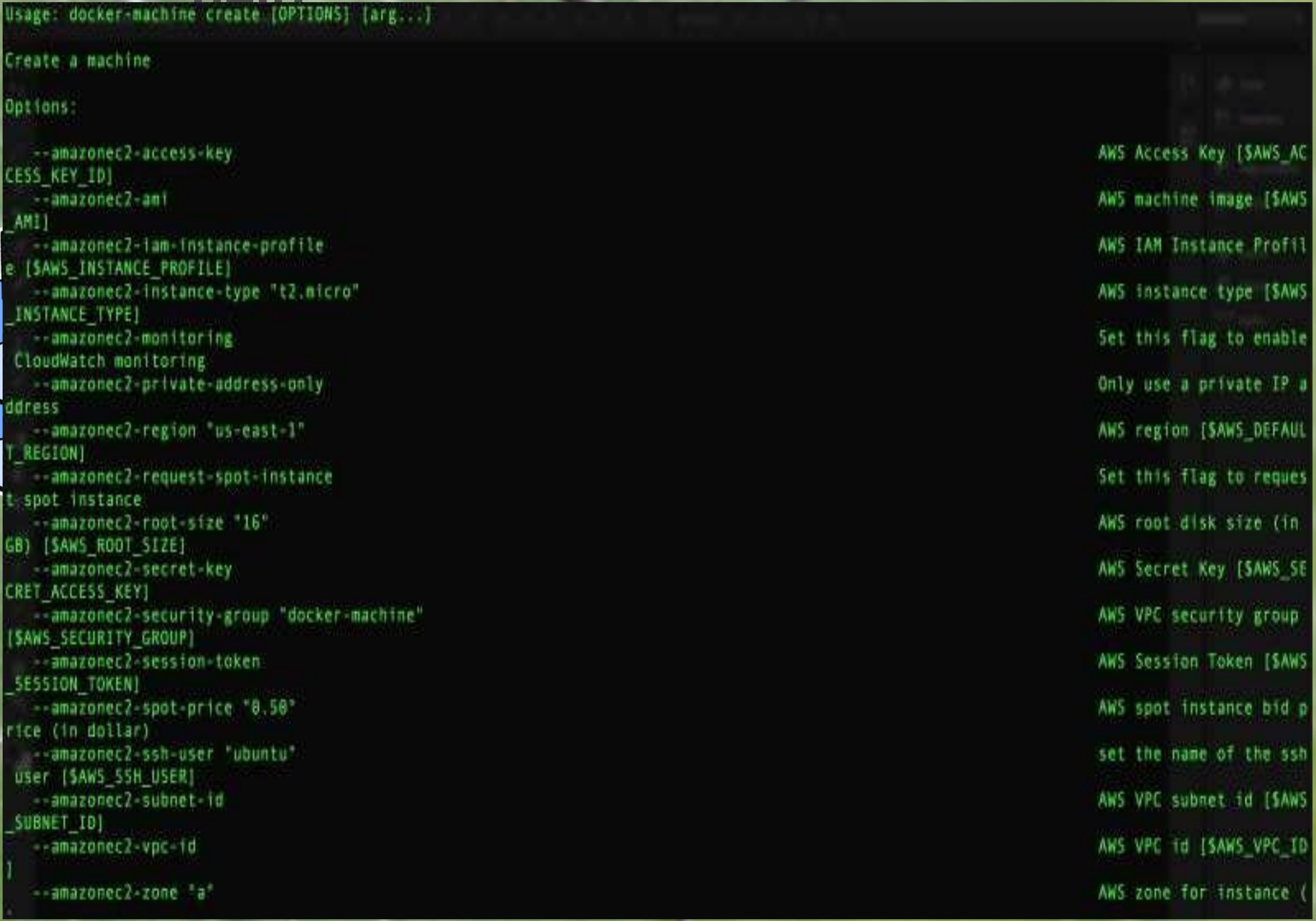
A bit real example

```
docker-machine create --driver
virtualbox  webapp
docker-machine ls
eval "$(docker-machine env
webapp)"  docker run -d -p 80:80
nginx
```

So what is this ---driver?

Docker-machine works with VMs and dockers inside it. Since the VMs can be on hosted servers, virtual or cloud, it provides drivers that lets to work with each one the way they require.

Diving to docker-machine create --help



```
Usage: docker-machine create [OPTIONS] [arg...]

Create a machine

Options:

   --amazonec2-access-key                                                                                    AWS Access Key [$AWS_AC
CESS_KEY_ID]
   --amazonec2-ami                                                                                           AWS machine image [$AWS
_AMI]
   --amazonec2-iam-instance-profile                                                                          AWS IAM Instance Profil
e [$AWS_INSTANCE_PROFILE]
   --amazonec2-instance-type "t2.micro"                                                                      AWS instance type [$AWS
_INSTANCE_TYPE]
   --amazonec2-monitoring                                                                                    Set this flag to enable
 CloudWatch monitoring
   --amazonec2-private-address-only                                                                          Only use a private IP a
ddress
   --amazonec2-region "us-east-1"                                                                            AWS region [$AWS_DEFAUL
T_REGION]
   --amazonec2-request-spot-instance                                                                         Set this flag to reques
t spot instance
   --amazonec2-root-size "16"                                                                                AWS root disk size (in
GB) [$AWS_ROOT_SIZE]
   --amazonec2-secret-key                                                                                    AWS Secret Key [$AWS_SE
CRET_ACCESS_KEY]
   --amazonec2-security-group "docker-machine"                                                               AWS VPC security group
[$AWS_SECURITY_GROUP]
   --amazonec2-session-token                                                                                 AWS Session Token [$AWS
_SESSION_TOKEN]
   --amazonec2-spot-price "0.50"                                                                             AWS spot instance bid p
rice (in dollar)
   --amazonec2-ssh-user "ubuntu"                                                                             set the name of the ssh
 user [$AWS_SSH_USER]
   --amazonec2-subnet-id                                                                                     AWS VPC subnet id [$AWS
_SUBNET_ID]
   --amazonec2-vpc-id                                                                                        AWS VPC id [$AWS_VPC_ID
]
   --amazonec2-zone "a"                                                                                      AWS zone for instance (
```

Yeah it’s very long

docker-machine create --help | wc -l

gives 157 options!!!

Filtering the option by driver

```
docker-machine create -d
<driver_name>
```

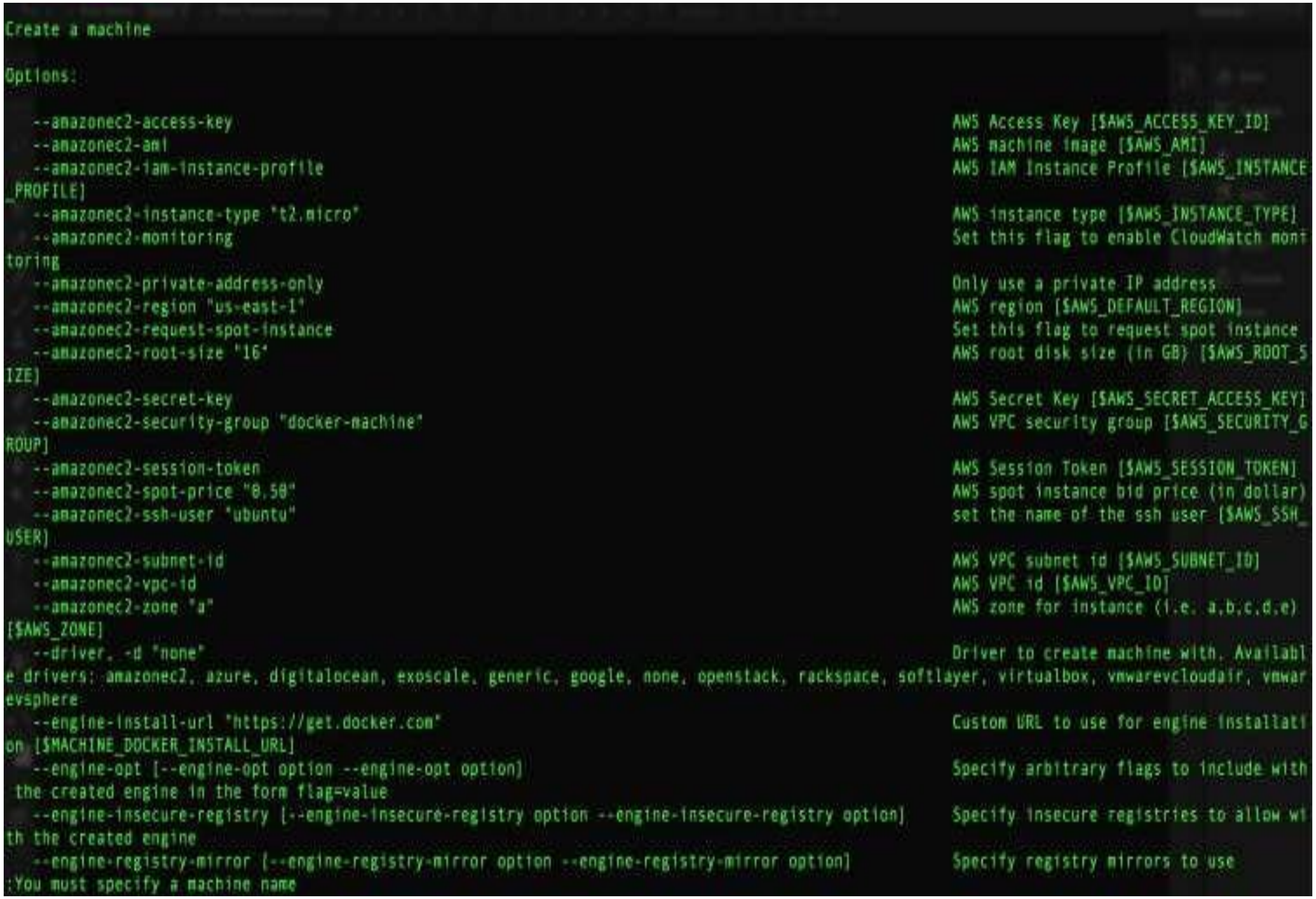
gives options only for specified driver filtering out unrelated options

Example: filter for AWS

```
docker-machine create -d amazonec2
```

Example: filter for AWS

docker-machine create -d amazonec2



```
Create a machine

Options:

   --amazonec2-access-key                                                     AWS Access Key [$AWS_ACCESS_KEY_ID]
   --amazonec2-ami                                                            AWS machine image [$AWS_AMI]
   --amazonec2-iam-instance-profile                                           AWS IAM Instance Profile [$AWS_INSTANCE
_PROFILE]
   --amazonec2-instance-type "t2.micro"                                       AWS instance type [$AWS_INSTANCE_TYPE]
   --amazonec2-monitoring                                                     Set this flag to enable CloudWatch moni
toring
   --amazonec2-private-address-only                                           Only use a private IP address
   --amazonec2-region "us-east-1"                                             AWS region [$AWS_DEFAULT_REGION]
   --amazonec2-request-spot-instance                                          Set this flag to request spot instance
   --amazonec2-root-size "16"                                                 AWS root disk size (in GB) [$AWS_ROOT_S
IZE]
   --amazonec2-secret-key                                                     AWS Secret Key [$AWS_SECRET_ACCESS_KEY]
   --amazonec2-security-group "docker-machine"                                AWS VPC security group [$AWS_SECURITY_G
ROUP]
   --amazonec2-session-token                                                  AWS Session Token [$AWS_SESSION_TOKEN]
   --amazonec2-spot-price "0.50"                                              AWS spot instance bid price (in dollar)
   --amazonec2-ssh-user "ubuntu"                                              set the name of the ssh user [$AWS_SSH_
USER]
   --amazonec2-subnet-id                                                      AWS VPC subnet id [$AWS_SUBNET_ID]
   --amazonec2-vpc-id                                                         AWS VPC id [$AWS_VPC_ID]
   --amazonec2-zone "a"                                                       AWS zone for instance (i.e. a,b,c,d,e)
[$AWS_ZONE]
   --driver, -d "none"                                                        Driver to create machine with. Availabl
e drivers: amazonec2, azure, digitalocean, exoscale, generic, google, none, openstack, rackspace, softlayer, virtualbox, vmwarevcloudair, vmwar
evsphere
   --engine-install-url "https://get.docker.com"                              Custom URL to use for engine installati
on [$MACHINE_DOCKER_INSTALL_URL]
   --engine-opt [--engine-opt option --engine-opt option]                     Specify arbitrary flags to include with
 the created engine in the form flag=value
   --engine-insecure-registry [--engine-insecure-registry option --engine-insecure-registry option]     Specify insecure registries to allow wi
th the created engine
   --engine-registry-mirror [--engine-registry-mirror option --engine-registry-mirror option]          Specify registry mirrors to use
:You must specify a machine name
```

Lab work: Create docker on digitalocean

Create VM with running nginx on digitalocean using docker-machine help

Access token:

<access token>

Remove those machines, NOW!

```
docker-machine rm
                <machine_name>
                docker-machine ls
```

What about connecting to physical servers?

To connect to local physical or virtual servers there are two ways:

1) by creating driverless VM
2) by creating using generic driver

For the first option there needs to be takes additional steps where you need to
create CA certificates using OpenSSL by following article written here
https://docs.docker.com/articles/https/

For the second option what is required is to put your public keys on physical or
virtual server

Lab work: Create generic VM to connect to server

Using docker-machine help create vm with generic driver

Host IP:
192.168.10.112

What about docker remote?

```
export
DOCKER_HOST=tcp://<remote_id>:<port>
docker run -d -p 80:80 nginx
```

Quiz

What is the difference between docker and docker- machine?

1) no difference. docker has remote api
2) docker is the client of docker-machine
3) docker-machine organizes vm and manages dockers
   inside
4) docker-machine is the manager of cloud vm for dockers

Quiz

What is the difference between docker and docker- machine?

1) no difference. docker has remote api
2) docker is the client of docker-machine
3) docker-machine organizes vm and manages dockers inside
4) docker-machine is the manager of cloud vm for dockers

Describe the ways of creating docker VMs to connect to host machines.

Quiz

Quiz

How many dockers can I create inside VM created by docker-machine?

Quiz

What is the difference between VM created with --generic driver and --virtualbox driver?

1. No difference. Difference only in driver names
2. IPs are different
3. After creating generic requires SSH access, whereas virtualbox SSH access is generated by docker-machine
4. When you remove virtualbox VM it removes all the data files, whereas generic removes only vm