docker-machine ssh

*Estimated reading time: 3 minutes*

Log into or run a command on a machine using SSH.

To login, just run docker-machine ssh machinename:

$ docker-machine ssh dev

## .

## ## ## ==

## ## ## ## ===

/""""""""""""""""\\_\_\_/ ===

~~~ {~~ ~~~~ ~~~ ~~~~ ~~ ~ / ===- ~~~

\\_\_\_\_\_\_ o \_\_/

\ \ \_\_/

\\_\_\_\_\\_\_\_\_\_\_/

\_ \_ \_\_\_\_ \_ \_

| |\_\_ \_\_\_ \_\_\_ | |\_|\_\_\_ \ \_\_| | \_\_\_ \_\_\_| | \_\_\_\_\_ \_ \_\_

| '\_ \ / \_ \ / \_ \| \_\_| \_\_) / \_` |/ \_ \ / \_\_| |/ / \_ \ '\_\_|

| |\_) | (\_) | (\_) | |\_ / \_\_/ (\_| | (\_) | (\_\_| < \_\_/ |

|\_.\_\_/ \\_\_\_/ \\_\_\_/ \\_\_|\_\_\_\_\_\\_\_,\_|\\_\_\_/ \\_\_\_|\_|\\_\\_\_\_|\_|

Boot2Docker version 1.4.0, build master : 69cf398 - Fri Dec 12 01:39:42 UTC 2014

docker@boot2docker:~$ ls /

Users/ dev/ home/ lib/ mnt/ proc/ run/ sys/ usr/

bin/ etc/ init linuxrc opt/ root/ sbin/ tmp var/

You can also specify commands to run remotely by appending them directly to the docker-machine ssh command, much like the regular ssh program works:

$ docker-machine ssh dev free

total used free shared buffers

Mem: 1023556 183136 840420 0 30920

-/+ buffers: 152216 871340

Swap: 1212036 0 1212036

Commands with flags will work as well:

$ docker-machine ssh dev df -h

Filesystem Size Used Available Use% Mounted on

rootfs 899.6M 85.9M 813.7M 10% /

tmpfs 899.6M 85.9M 813.7M 10% /

tmpfs 499.8M 0 499.8M 0% /dev/shm

/dev/sda1 18.2G 58.2M 17.2G 0% /mnt/sda1

cgroup 499.8M 0 499.8M 0% /sys/fs/cgroup

/dev/sda1 18.2G 58.2M 17.2G 0%

/mnt/sda1/var/lib/docker/aufs

If you are using the “external” SSH type as detailed in the next section, you can include additional arguments to pass through to the ssh binary in the generated command (unless they conflict with any of the default arguments for the command generated by Docker Machine). For instance, the following command will forward port 8080 from the default machine to localhost on your host computer:

$ docker-machine ssh default -L 8080:localhost:8080

Different types of SSH

When Docker Machine is invoked, it will check to see if you have the venerable ssh binary around locally and will attempt to use that for the SSH commands it needs to run, whether they are a part of an operation such as creation or have been requested by the user directly. If it does not find an external ssh binary locally, it will default to using a native Go implementation from [crypto/ssh](https://godoc.org/golang.org/x/crypto/ssh). This is useful in situations where you may not have access to traditional UNIX tools, such as if you are using Docker Machine on Windows without having msysgit installed alongside of it.

In most situations, you will not have to worry about this implementation detail and Docker Machine will act sensibly out of the box. However, if you deliberately want to use the Go native version, you can do so with a global command line flag / environment variable like so:

$ docker-machine --native-ssh ssh dev

There are some variations in behavior between the two methods, so please report any issues or inconsistencies if you come across them.