Augustus 2021

Robert Hartevelt

Mirantis Presales Northern Europe

Installation Mirantis Demo Environment

# INDEX

Table of Contents

[INDEX 1](#_Toc80017010)

[Foreword 2](#_Toc80017011)

[Let’s get started 3](#_Toc80017012)

[Step 1: License file 3](#_Toc80017013)

[Step 2: Start bootstrapper AWS instance 3](#_Toc80017014)

[Step 3: Bootstrap user or not a bootstrap user – that is the question… 4](#_Toc80017015)

[Step 4: Connect to our EC2 instance 5](#_Toc80017016)

[Step 5: Prepare our EC2 instance – Run script 1 6](#_Toc80017017)

[Step 6: Bootstrapper user (only for new bootstrapper user – NOT skipped step 3). 6](#_Toc80017018)

[Step 7: Mirantis Container Cloud only or include a child Cluster 7](#_Toc80017019)

[Step 8: Mirantis Container Cloud installation 7](#_Toc80017020)

[Step 9: We made it 8](#_Toc80017021)

[Step 10: What do we have 9](#_Toc80017022)

# Foreword

Mirantis wants to show you how Mirantis Container Cloud and Mirantis Kubernetes Environment looks like and works. Therefore, we made it easier for everybody to install this combination.

Benefits for Mirantis Container Cloud:

* Monitoring, Logging and Alerting is directly included and working = Second day Operations
* Complete Kubernetes Environment which handles K8s and Swarm workload.
* On whatever infrastructure you place Container Cloud and/or Kubernetes of Mirantis, it always is the same (read: easy migrations of workload between different infra (AWS, Azure, Bare-metal, VMWare or OpenStack).

Back on the ground this is what this document is about:

* We will start an Ubuntu Linux instance on AWS
* Configure how the AWS CLI is accessed and in which region we are building it
* Run the script which will build an Mirantis Container Cloud and Kubernetes environment.

It will guide you through the technical details.

# Let’s get started

## Step 1: License file

Get your free demo license.

Goto <https://www.mirantis.com/download/mirantis-cloud-native-platform/mirantis-container-cloud/>

Register and download the mirantis.lic file

## Step 2: Start bootstrapper AWS instance

Log into your AWS account (AWS Management Console)

Goto EC2

Click ‘Launch Instances’

In the ‘Search for an AMI by entering a search term’ type : Ubuntu

Choose for Ubuntu Server 18.04(HVM, SSD Volume Type – 64 bit (X86)

Use at least an t3.2xlarge Instance type

Click ‘Next Configure Instance Details’

Click ‘Next: Add Storage’

Change the Size of the Root storage to 20 GB.

In Tags you may add some Tags so we can find your bootstrapper more easily.

Add Tag: Key: Name. Value: Kaas-bootstrapper (or any other name you like….).

Review and Launch + Lauch

Create a new key pair (+download) or use a key pair where you hold the primary key of!

Acknowledge you have the keys and Launch the Instance.

You Instance should be created in the DEFAULT public Virtual Private Cloud (VPC).

## Step 3: Bootstrap user or not a bootstrap user – that is the question…

If you work at Mirantis and hold an already existing bootstrap user in IAM goto Step 4.

You are not already having a bootstrap user, let’s create one.

In order to let the first script, create the bootstrap user the just created EC2 instance (Step 2) needs a role with Admin rights.

Please follow along:

Go on AWS Management Console to IAM:

* Goto Roles
* Create Role
  + Click ‘AWS Service’ block
  + Click ‘EC2’ (underneath Common Use cases).
  + Click ‘Next: Permissions’
  + In the search bar type: ‘Admin’
  + Tick the box(!) in front of ‘AdministratorAccess’
  + Click ‘Next: Tags’
  + Add tags: Key: Name Value: Something\_so\_we\_can\_find\_it\_back…
  + Click ‘Next: Review’
  + Fill Role name: Admin2bootstrapper (or any other name we can find back…)
  + Click ‘create role’

Now head back to EC2:

* Type in ‘Search for services, features, …’ EC2
* Tick the box (!) in front of your EC2 instance created in step 2
* Goto ‘Actions (Right top), Security, Modify IAM role
* IAM role: Find your just created IAM role and add that to the EC2 instance of step 2.
* Click ‘Save’.

## Step 4: Connect to our EC2 instance

Connect with your EC2 instance using your private key (User = Ubuntu). IP address can be found in the Instance details (public IP).

ssh ubuntu@<IP> -i <private-key-file>

Let’s download our script material.

wget <https://mirantis-emea-demo-environment.s3.eu-central-1.amazonaws.com/create_demo_env1.0.tar>

Unpack:

tar xvf create\_demo\_env1.0.tar

You should now have a directory called demo\_1.0

cd demo\_1.0/

IF you skipped Step 3 because you already have a bootstrapper user ONLY then:

vi sourceme.ksh

Change (cw = change word) FILL-IN with your bootstrapper ACCESS\_KEY & SECRET\_KEY.

Esc+ :wq writes the file to disk.

You may want to also change the AWS REGION in sourceme.ksh to your liking…

**One more thing: Get your license file in place**

Create a file having the licenses information gathered in step 1.

cd demo\_1.0/

vi mirantis.lic. 🡪 Put your license.lic file information in here and save

Or upload your mirantis.lic file to the demo\_1.0/ directory

## Step 5: Prepare our EC2 instance – Run script 1

It is time to run script one.

Script one will:

* Update the host with the latest and greatest packages
* Install docker
* Install AWSCLI
* Configure AWSCLI config.
* Install the Mirantis Cloud Container installation directory
* Create a bootstrapper user (if sourceme.ksh is untouched).

cd demo\_1.0/

./1\_prepare.ksh

……….

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Now IMPORTANT STUFF!!!: \*\*

\*\* - fill in the correct KEYS + SECRET in sourceme.ksh \*\*

\*\* - Remove the admin IAM role from the EC2 instance \*\*

\*\* - Logout! \*\*

\*\* - Log back in - continue running the second script \*\*

## Step 6: Bootstrapper user (only for new bootstrapper user – who did NOT skip step 3).

We have created a special bootstrap user which now needs to be connected to the scripting.

Goto the AWS Management Console:

* Searchbar for services, features (on-top) type : IAM
* Goto Users

You will find a new user called bootstrapper.cluster-api-provider-aws.kaas.mirantis.com

* Click on the Username link (blue)
* Click ‘Security credentials’
* Click ‘Create access key’
* Either download .csv file or copy both credentials (only showed once!) NOW.

Go back into your EC2-instance and modify the sourceme.ksh script.

vi sourceme.ksh

Change (cw = change word) FILL-IN with your bootstrapper ACCESS\_KEY & SECRET\_KEY.

Esc+ :wq writes the file to disk.

**You can also modify the AWS REGION where the installation will be in.**

The, remove the Admin IAM role from the EC2-instance

Goto the AWS Management Console:

* Searchbar for services, features (on-top) type: EC2
* Tick the box (!) in front of your EC2 instance created in step 2
* Goto ‘Actions (Right top), Security, Modify IAM role
* IAM role: Change this to ‘No IAM Role’
* Click ‘Save’.

**Important!**

Your brand new EC2 instances has installed docker on your system and has granted the ubuntu user access (by adding it to its group). In order to have the effect in place, you will need:

* Log out of the EC2 Instance
* Log back in to the EC2 instance as Ubuntu.

## Step 7: Mirantis Container Cloud only or include a child Cluster

The installation will create a Kubernetes (k8s) cluster of three nodes which will handle or monitoring, logging and self-service activities. This is NOT meant to run any k8s workloads!

A child cluster is a K8s cluster consisting of 3 workers and 3 master nodes which is dedicated to k8s workload.

If you only want Mirantis Container Cloud service for now, and manually later add K8s services to this, you can have a faster install (~30minuten) by deleting/renaming the CREATE\_A\_MKECHILD\_PLEASE file

## Step 8: Mirantis Container Cloud installation

Let’s finishing off…

cd demo\_1.0/

./2\_mcc\_mke\_setup.ksh

SOURCE sourceme.ksh

nohup: appending output to 'nohup.out'

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Started building Mirantis Container Cloud - Happening in the background \*\*

\*\* - Process can be followed by tailing kaas-bootstrap/nohop.out - \*\*

\*\* This process will take about 40 minutes \*\*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

|

..

Cluster creation wil be done in the background and will take about ~40 minutes or so…

A **.** appears every minute and should be done when it gets close to the **|** sign.

The script will however continue until the cluster creation is done. Which may be never if it encounters real issues….

You can take another terminal can follow nohup.out where the real output of the Mirantis Container Cloud installation is.

tail -f kaas-bootstrap/nohup.out

If you follow this output, you will see fails, errors, e.g., because the cluster creations try to connect to services which are not (yet) there. This is NORMAL behavior.

## Step 9: We made it

++++++++++++++ END GOOD - ALL GOOD +++++++++++++++++++++++++++++++++

----------------------------

passwords:

iamDBAdmin: BwnualKckanGWq

iamDBSst: VgG5PNHyTYYmxq

keyCloak: 9rCdNVE2xNiWAG

I have already created a user for you:

Username: demo

password: yes, that is IT (read: The password is 'password')

---------------------------------------------------------------------

Connection DETAILS:

MCCURL=http://a7d2807a5f45c4f73804bcfaab520b15-120609440.eu-west-1.elb.amazonaws.com

KEYLOAKURL=https://a48914db16a02417e8f19def1f02414a-384231793.eu-west-1.elb.amazonaws.com

Keycloak user + password = keyCloak + keyCloak: 9rCdNVE2xNiWAG

-----------------------------------------------------------

This intel will be saved in todays\_MCC\_servers.logfile

+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++

**This information is just an example, and the cluster is NO longer available.**

## Step 10: What do we have

**Mirantis Container Cloud:**

In order to connect to Mirantis Container Cloud you can connect to the MCCURL mentioned website.

Yes, you may will need (~3 x) to Advanced and proceed to the website. The certificates are self-signed and need to be established for 3 (master) hosts.

In Mirantis Container Cloud you will have a default project which ssh-keys and AWS credentials (of the bootstrapper user) already in place.

You can create a new child Mirantis Kubernetes Engine (MKE) environment right from here.

Create an old version and then look at the ‘iPad’ like upgrade… And/or test workload. Please mind that the test license can only build 3 cluster having 6 nodes each.

**Mirantis Kubernetes Engine (MKE):**

When the installation also installed the first Child MKE cluster you will find it in Mirantis Container Cloud.

Sign into Mirantis Container Cloud IAM with the user demo / password

Click on the (blue) Demo name to enter the details of the demo Cluster.

From here you can go into MKE by clicking the MKE GUI or see the monitoring of MCC by clicking on Grafana UI.

When connection to the MKE GUI you are asked to give in a user/password or use keyCloak as your login. Use keycloak.

**KeyCloak:**

You can create your own users / password at the KEYLOAKURL mentioned website.

Then use the keyCloak user with the password mentioned after keyCloak:

**Thank you:**

We hope you enjoy working with Mirantis Container Cloud and Mirantis Kubernetes Engine.

Please mind we will always have the same result of k8s regardless of underlaying infrastructure. We now have done the installation on AWS, but the same MCC/MKE result can be accomplished on bare-metal, VMWare, Equinix and Azure & Google cloud (last 2 on day of writing on the roadmap).

And please ensure the complete package of k8s which includes second day (Operations) Monitoring, Logging and Alerting out-of-the-box.

Kind Regards,

EMEA PreSales team