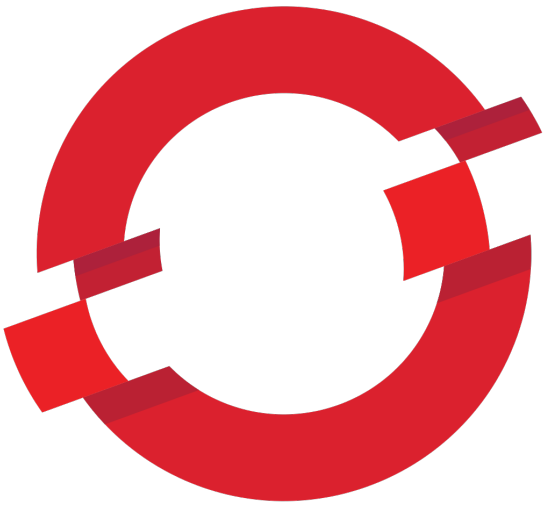


OPENSHIFT

OpenShift Installation using Ansible & Terraform

DOCUMENT VERSION V 1.0

Authors

Adam Muganwa
Linux Systems Engineer
adam.muganwa@ibm.com
Slack: Adam Muganwa

Ben Rossouw
IT Specialist & Infrastructure Manager
brossouw@za.ibm.com
Slack: Ben Rossouw

Openshift 4x Installation vSphere

Prerequisites

Define OCP cluster id (cluster name) I will be using **envisage**

User-provisioned DNS requirements
https://docs.openshift.com/container-platform/4.2/installing/installing_vsphere/installing-vsphere.html#installation-dns-user-infra_installing-vsphere

Setup Bastion node on the network.

Bastion Node used in this documentation is a Centos VM

Cleaning up previous installation
In installation folder
rm -rf': rm -rf auth -R *.ign .openshift_install* metadata.json

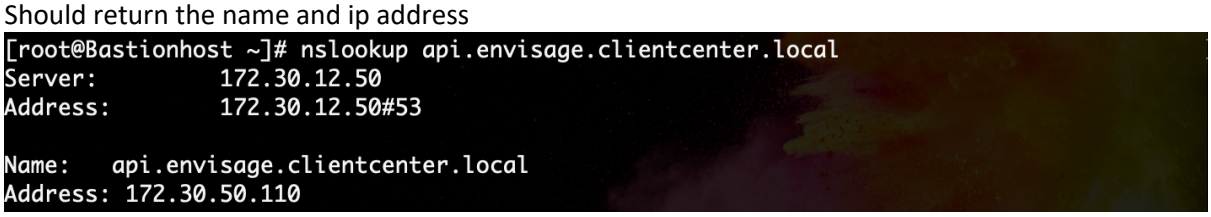
In vsphere floder
rm -rf terraform.tfstate .terraform/ -R

Installing required packages

yum install ansible -y
yum install git -y
yum install wget unzip -y
yum install bind-utils -y
yum install vim -y

Check DNS entries
nslookup api.envisage.clientcenter.local

Should return the name and ip address



Check the rest of the records.
nslookup api-int.envisage.clientcenter.local
nslookup *.apps.envisage.clientcenter.local
nslookup bootstrap-0.envisage.clientcenter.local
nslookup control-plane-0.envisage.clientcenter.local
nslookup control-plane-1.envisage.clientcenter.local
nslookup control-plane-2.envisage.clientcenter.local
nslookup compute-0.envisage.clientcenter.local
nslookup compute-1.envisage.clientcenter.local
nslookup compute-2.envisage.clientcenter.local

nslookup compute-3.envisage.clientcenter.local
nslookup compute-4.envisage.clientcenter.local

nslookup -type=SRV _etcd-server-ssl._tcp.envisage.clientcenter.local

```
[root@Bastionhost ~]# nslookup -type=SRV _etcd-server-ssl._tcp.envisage.clientcenter.local
Server:         172.30.12.50
Address:        172.30.12.50#53

_etcd-server-ssl._tcp.envisage.clientcenter.local       service = 0 10 2380 etcd-2.envisage.clientcenter.local.
_etcd-server-ssl._tcp.envisage.clientcenter.local       service = 0 10 2380 etcd-1.envisage.clientcenter.local.
_etcd-server-ssl._tcp.envisage.clientcenter.local       service = 0 10 2380 etcd-0.envisage.clientcenter.local.
```

Setup Load Balancer

Download installer files
git clone https://github.com/adamsocool/Openshift4x-Installation

copy the downloaded loadbalancer.yml
sudo cp loadbalancer.yml /etc/ansible

Set up Load balancer/http server using playbook
cd /etc/ansible
edit hosts file and add your load balancer to the file
vim hosts

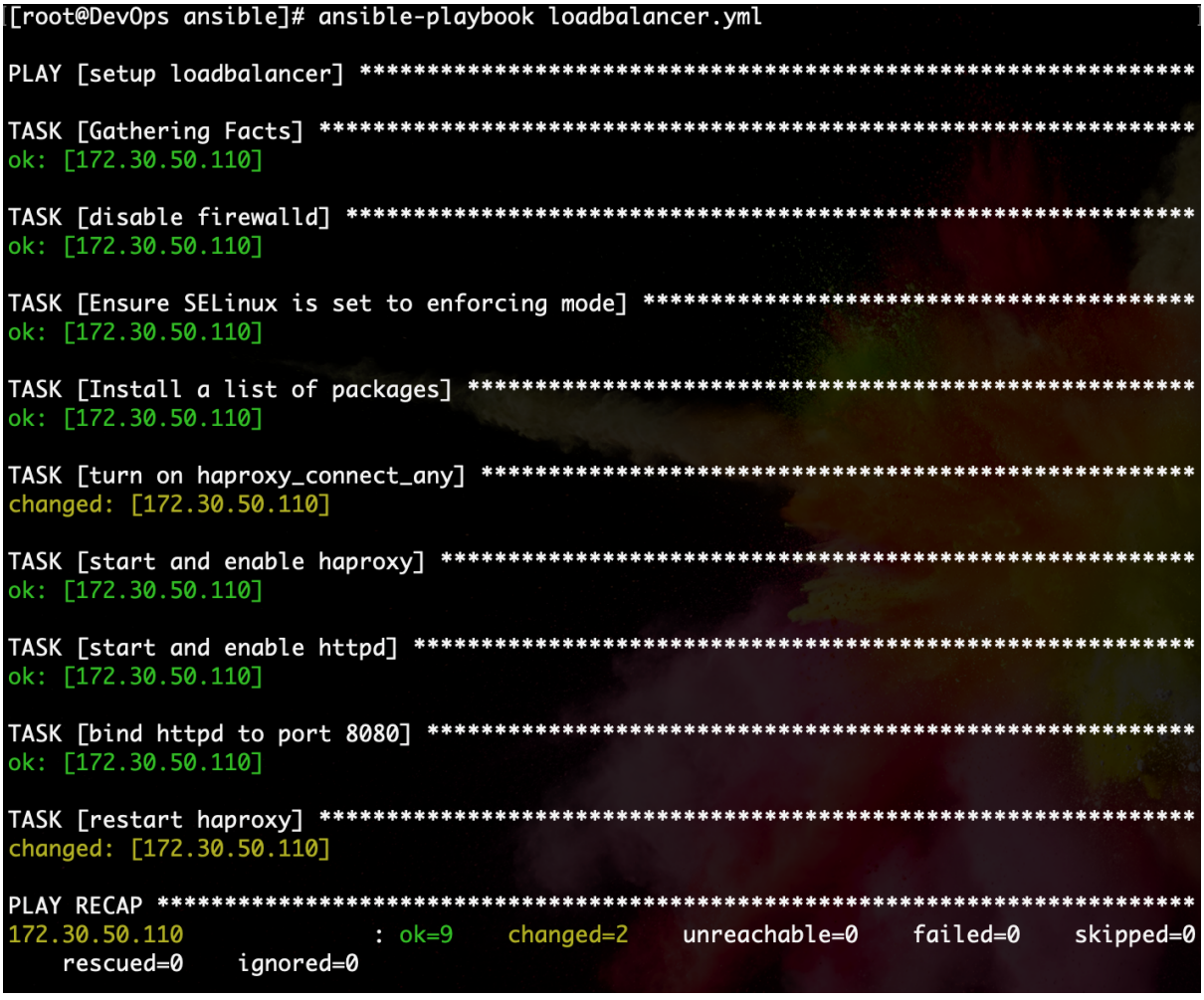
```
[loadbalancer]
172.30.50.110
~
```

generate ssh-key and copy to loadbalancer
ssh-keygen
ssh-copy-id root@172.50.30.110

Check that you can connect to the server using ansible
ansible -m ping loadbalancer -o

```
[root@DevOps ansible]# ansible -m ping loadbalancer -o
172.30.50.110 | SUCCESS => {"ansible_facts": {"discovered_interpreter_python": "/usr/lib
exec/platform-python"},"changed": false,"ping": "pong"}
[root@DevOps ansible]#
```

Run the playbook
ansible-playbook loadbalancer.yml



Login to Load Balancer

This configuration will add the following load balancer entries to HA-Proxy:
openshift-api-server *(port 6443)*
machine-config-server *(port 22623)*
ingress-http *(port 80)*
ingress-https *(port 443)*
Add the following configuration lines to /etc/haproxy/haproxy.cfg:
(replace the IP addresses!)

cd /etc/haproxy

Download haproxy.cfg file

backup haproxy file
cp haproxy.cfg haproxy.cfg.bk

Replace the haproxy.cfg file with the one in Openshift4x-Installation folder
vim haproxy.cfg

Add the following configuration lines to /etc/haproxy/haproxy.cfg:
(replace the IP addresses!)

```
    maxconn                 3000

frontend openshift-api-server
    bind *:6443
    use_backend openshift-api-server
    mode tcp
    option tcplog

backend openshift-api-server
    balance source
    mode tcp
    server bootstrap-0 172.30.50.70:6443 check
    server control-plane-0 172.30.50.71:6443 check
    server control-plane-1 172.30.50.72:6443 check
    server control-plane-2 172.30.50.73:6443 check

frontend machine-config-server
    bind *:22623
    default_backend machine-config-server
    mode tcp
    option tcplog

backend machine-config-server
    balance source
    mode tcp
    server bootstrap-0 172.30.50.70:22623 check
    server control-plane-0 172.30.50.71:22623 check
    server control-plane-1 172.30.50.72:22623 check
    server control-plane-2 172.30.50.73:22623 check

frontend ingress-http
    bind *:80
    default_backend ingress-http
    mode tcp
    option tcplog
```

```
backend ingress-https
    balance source
    mode tcp
    server compute-0 172.30.50.74:443 check
    server compute-1 172.30.50.75:443 check
    server compute-2 172.30.50.76:443 check
    server compute-3 172.30.50.77:443 check
    server compute-4 172.30.50.78:443 check
```

Reboot server

Log Back into Bastion node

Create core user

useradd core
passwd core
usermod -aG wheel core

Swtich to core user

su - core

Downloading required files

mkdir install_ocp4
cd install_ocp4
Install terraform version 0.11.14
wget https://releases.hashicorp.com/terraform/0.11.14/terraform_0.11.14_linux_amd64.zip
unzip terraform_0.11.14_linux_amd64.zip
sudo cp terraform /usr/local/bin
terraform -v
Will return the version of terraform
Terraform v0.11.14

copy terraform installer form the downloaded repo
sudo cp /root/Openshift4x-Installation/vsphere -R /home/core/install_ocp4

Download ocp installer and client
wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.3.1/openshift-install-linux-4.3.1.tar.gz

wget https://mirror.openshift.com/pub/openshift-v4/clients/ocp/4.3.1/openshift-client-linux-4.3.1.tar.gz

tar -xvf openshift-install-linux-4.3.1.tar.gz
tar -xvf openshift-client-linux-4.3.1.tar.gz

Download OVA image and upload to vSphere and convert OVA image to template
https://mirror.openshift.com/pub/openshift-v4/dependencies/rhcos/
Select the version you want to install

Since our CoreOS based VMs will be configured automatically, we have to provide a ssh public key to be able to log on to the nodes via ssh as user core
ssh-keygen -t rsa -b 4096 -N ''

start ssh-agent
eval "$(ssh-agent -s)"

add your ssh private key to the ssh-agent
ssh-add ~/.ssh/id_rsa

For installing Red Hat OpenShift Container Platform 4x, you'll need a pull secret:
Link: https://cloud.redhat.com/openshift/install/vsphere/user-provisioned

copy your pull secret and save it in a text file

copy install-config.yaml from Openshift4x-Installation folder
cp install-config.yaml install-config.yaml.bk
vim install-config.yaml

enter your vcenter details, pull secret and ssh-key.

```
apiVersion: v1
baseDomain: clientcenter.local
compute:
- hyperthreading: Enabled
  name: worker
  replicas: 5
controlPlane:
  hyperthreading: Enabled
  name: master
  replicas: 3
metadata:
  name: envisage
networking:
  clusterNetworks:
  - cidr: 10.254.0.0/16
    hostPrefix: 24
  networkType: OpenShiftSDN
  serviceNetwork:
  - 192.168.0.0/16
platform:
  vSphere:
    vcenter: labvcsa.clientcentre.local
    username: administrator@vsphere.local
    password:
    datacenter: Datacenter01
    defaultDatastore: DsVol01
pullSecret: '{"auths":{"cloud.openshift.com":{"auth":"b3BlbnNoaWZ0LXJlbGVhc2UtZGV2K2FkYW1
NEFHMTUyME9RTVBXQjVEU0kzV0c3RjJMSktJVzNXNTdTVTYxQTI=","email":"adam.muganwa@ibm.com"},"qu
WZiY2txaHlkc2loOkMzRFEyRlYxUTJYRUI2WFE3U1IyNk4yQzRTNEFHMTUyME9RTVBXQjVEU0kzV0c3RjJMSktJVz
I3NDk2Nzd8dWhjLTFUcG5VbGVrZTdyRHZJYnlmYkNrUUhZRFNpaDpleUpoYkdjaU9pSlNVelV4TWlKOS5leUp6ZFd
d2d4YmRHaG5YNkF0c2lMYnAtNjFBOWtjNDRHNURRVWhETWxxajBybWVvS1l3LWZLWkcxMlYwX2dzSlU1VjhlMHB1R
g0NGFkS3FVbmZXWUx0TFRoTnNoLWVwcTB3aUVuOVdCUTVTZTJ5eHdHU3R3bE9icGJsd2pxT2l5c21WTlpsNTNael9
a1lPVS12NUhXcWZpVHowUXVBWGtZNkRBR2NSZzUwZjI1cm1fSkFsb3h3d1BTcXBGb194RXhGcXQ1ODBPRThZNDRjY
BPd0poZ2d2NFJ6bTRfcGpaNXhXMzVldXZES3BPcUV2dnp1d0JtMVNOUkdZRmJkQ2UzOWZvWlFtaXZIZlFNQ3VlVU1
NkNORkxucmFaMTI5eXQzSjdBdjBaWmFkdVRhRTY4dC1Rb3lmWU42X2lacjl2UGRBaXJ4VVE4cm5TXzU4emxES3hTU
==","email":"adam.muganwa@ibm.com"},"registry.redhat.io":{"auth":"NTI3NDk2Nzd8dWhjLTFUcG5
QmhPRFUwWWpGak9EZzFaVEEyT1Rka05tWm1OMkUxT0NKOS5UOWlKQUFPNnFxN1FEMWdCd2d4YmRHaG5YNkF0c2lMY
RYMG13QlJIRnU0WGNVT2FielRvUHRubzBLR2QwVzMxaWtyZlpnVlBjS1B3czJGa3BBcHg0NGFkS3FVbmZXWUx0TFR
VVdONVVpWjUxdkhxaWtlbWJNbnc2aElhS1lOWnp5NGx2M05maFJKVWtYN2xIblMxXzlma1lPVS12NUhXcWZpVHowU
lmeFZKSXRnS3lERzY4b2V2ekpzSE4yQ1g5a0tRNU5ZMFBuOWVrNXpLbjZqZ0tRUUR3ZFBPd0poZ2d2NFJ6bTRfcGp
cXZDZkc0UkZjVE81WTNPUTlfZUZObkVmaUJSWUtQbGhZd1paOE02a0kwZGlwUUVzOG51NkNORkxucmFaMTI5eXQzS
JpRzl5NHdNV2NHdE1aekV0cTZCR3VQMTVzNUZHM1dlTVpQbmphZVh5bFdERUJWOC1kRQ==","email":"adam.mug
sshKey: ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAACAQCiLEjFQ8gsYzYGa1A/6hH6zAe5P81fKrZPJl2LTo8Wg
@@@
-- INSERT --
```

create ignition configs
./openshift-install create ignition-configs

```
[core@Bastionhost install_ocp4]$ ./openshift-install create ignition-configs
INFO Consuming "Install Config" from target directory
```

ignition files are valid for 24 hours - so if your installation takes longer than 24 hours due to issues, you have to generate new ignition files.

Copy ignition files to your HTTP server

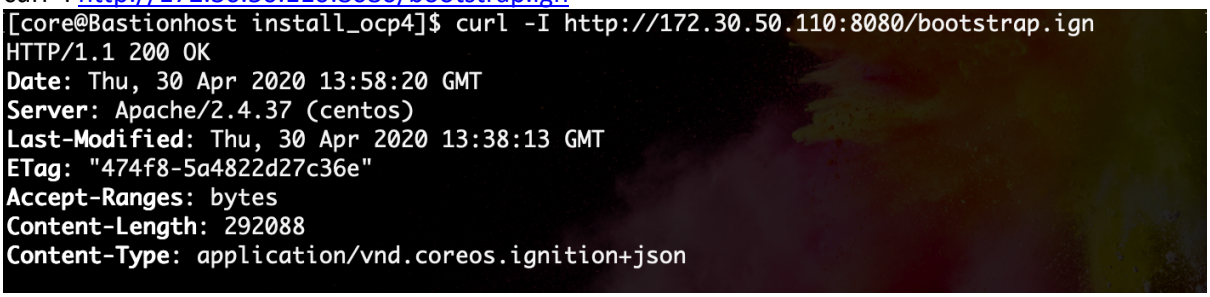
Copy the generated bootstrap.ign file to your HTTP server and ensure, that the file can be downloaded with http:
scp bootstrap.ign root@172.30.50.110:/var/www/html/

Login to Load Balancer

chown apache.apache /var/www/html/bootstrap.ign
chmod 666 /var/www/html/bootstrap.ign
systemctl restart httpd
systemctl restart haproxy

On Bastion

Check , if download would succeed from your http server.
curl -I http://172.30.50.110:8080/bootstrap.ign



edit the terraform installer
cd /vsphere

vim terraform.tfvars

In sections control_plane_ignition / END_OF_MASTER_IGNITION and compute_ignition / END_OF_WORKER_IGNITION,
insert / copy&paste the contents of the ignitions files (master and worker) we generated before:
END_OF_MASTER_IGNITION -> master.ign
END_OF_WORKER_IGNITION -> worker.ign

```
cluster_id = "envisage"
cluster_domain = "envisage.clientcenter.local"
base_domain = "clientcenter.local"
vsphere_server = "labvcsa.clientcentre.local"
vsphere_user = "administrator@vsphere.local"
vsphere_password = ""
vsphere_cluster = "Cluster01"
vsphere_datacenter = "Datacenter01"
vsphere_datastore = "DsVol01"
vm_template = "rhcos-4.2.0"
machine_cidr = "172.30.50.0/24"

vm_network = "VM Network"

control_plane_count = 3
compute_count = 5

bootstrap_ignition_url = "http://172.30.50.110:8080/bootstrap.ign"

control_plane_ignition = <<END_OF_MASTER_IGNITION
{"ignition":{"config":{"append":[{"source":"https://api-int.envisage.clientcenter.local:
22623/config/master","verification":{}}]},"security"....................
END_OF_MASTER_IGNITION

compute_ignition = <<END_OF_WORKER_IGNITION
{"ignition":{"config":{"append":[{"source":"https://api-int.envisage.clientcenter.local:
22623/config/worker","verification":{}}]},"security":{"tls":.................
END_OF_WORKER_IGNITION

# put your static IPs in here
bootstrap_ip = "172.30.50.70"
control_plane_ips = ["172.30.50.71", "172.30.50.72", "172.30.50.73"]
compute_ips = ["172.30.50.74", "172.30.50.75", "172.30.50.76", "172.30.50.77", "172.30.5
0.78"]


~
~
```

Modify the file main.tf and delete or comment out the module "dns" part, since we do not want to use Route53/AWS as DNS provider:

```
//module "dns" {
//  source = "./route53"
//
//  base_domain         = "${var.base_domain}"
//  cluster_domain      = "${var.cluster_domain}"
//  bootstrap_count     = "${var.bootstrap_complete ? 0 : 1}"
//  bootstrap_ips       = ["${module.bootstrap.ip_addresses}"]
//  control_plane_count = "${var.control_plane_count}"
//  control_plane_ips   = ["${module.control_plane.ip_addresses}"]
//  compute_count       = "${var.compute_count}"
//  compute_ips         = ["${module.compute.ip_addresses}"]
//}
                                                              101,1         Bot
```

configure DNS and network gateway, if needed

vim vsphere/machine/ignition.tf

```
  content {
    content = <<EOF
TYPE=Ethernet
BOOTPROTO=none
NAME=ens192
DEVICE=ens192
ONBOOT=yes
IPADDR=${local.ip_addresses[count.index]}
PREFIX=${local.mask}
GATEWAY=${local.gw}
DOMAIN=${var.cluster_domain}
DNS1=172.30.12.50
DNS2=9.9.9.9
EOF
  }
}
```

Set disk size

vim vsphere/machine/main.tf

```
  network_interface {
    network_id = "${data.vsphere_network.network.id}"
  }

  disk {
    label            = "disk0"
    size             = 150
    thin_provisioned = "${data.vsphere_virtual_machine.template.disks.0.thin_provisioned
}"
```

In the vSphere installer directory

terraform init

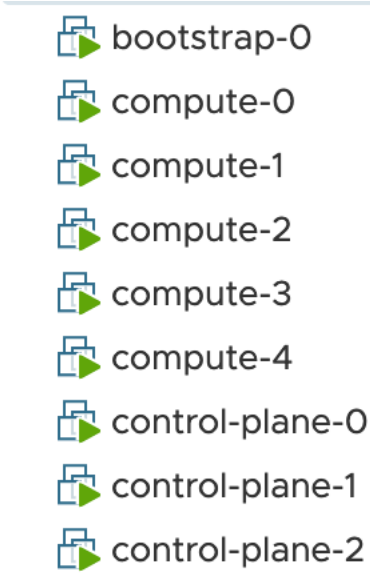
```
Terraform has been successfully initialized!

You may now begin working with Terraform. Try running "terraform plan" to see
any changes that are required for your infrastructure. All Terraform commands
should now work.

If you ever set or change modules or backend configuration for Terraform,
rerun this command to reinitialize your working directory. If you forget, other
commands will detect it and remind you to do so if necessary.
[core@Bastionhost vsphere]$
```

terraform plan

```
  + module.folder.vsphere_folder.folder
      id:                                                   <computed>
      datacenter_id:                                        "datacenter-2"
      path:                                                 "envisage"
      type:                                                 "vm"

  + module.resource_pool.vsphere_resource_pool.resource_pool
      id:                                                   <computed>
      cpu_expandable:                                       "true"
      cpu_limit:                                            "-1"
      cpu_reservation:                                      "0"
      cpu_share_level:                                      "normal"
      cpu_shares:                                           <computed>
      memory_expandable:                                    "true"
      memory_limit:                                         "-1"
      memory_reservation:                                   "0"
      memory_share_level:                                   "normal"
      memory_shares:                                        <computed>
      name:                                                 "envisage"
      parent_resource_pool_id:                              "resgroup-8"


Plan: 11 to add, 0 to change, 0 to destroy.

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

Note: You didn't specify an "-out" parameter to save this plan, so Terraform
can't guarantee that exactly these actions will be performed if
"terraform apply" is subsequently run.
```

terraform apply -auto-approve

You should now see VMs in your vCenter



```
module.compute.vsphere_virtual_machine.vm[1]: Creation complete after 19s (ID: 4222e771-4
module.bootstrap.vsphere_virtual_machine.vm: Creation complete after 19s (ID: 42227317-c1
module.compute.vsphere_virtual_machine.vm[4]: Creation complete after 19s (ID: 42222298-b
module.compute.vsphere_virtual_machine.vm.3: Still creating... (20s elapsed)
module.control_plane.vsphere_virtual_machine.vm.0: Still creating... (20s elapsed)
module.control_plane.vsphere_virtual_machine.vm.2: Still creating... (20s elapsed)
module.compute.vsphere_virtual_machine.vm.0: Still creating... (20s elapsed)
module.compute.vsphere_virtual_machine.vm[2]: Creation complete after 20s (ID: 4222bad9-5
module.compute.vsphere_virtual_machine.vm[0]: Creation complete after 20s (ID: 4222a1f8-8
module.control_plane.vsphere_virtual_machine.vm[0]: Creation complete after 21s (ID: 4222
module.compute.vsphere_virtual_machine.vm[3]: Creation complete after 22s (ID: 42224f56-7
module.control_plane.vsphere_virtual_machine.vm[2]: Creation complete after 23s (ID: 4222

Apply complete! Resources: 11 added, 0 changed, 0 destroyed.
[core@bastion-prod vsphere]$
```

cd install_ocp4 folder

./openshift-install --dir=. wait-for bootstrap-complete --log-level debug

```
[core@bastion-prod ocpinstall]$ openshift-install --dir=. wait-for bootstrap-complete --log-
DEBUG OpenShift Installer v4.3.5
DEBUG Built from commit 82f9a63c06956b3700a69475fbd14521e139aa1e
INFO Waiting up to 30m0s for the Kubernetes API at https://api.envisage.clientcenter.local:6
DEBUG Still waiting for the Kubernetes API: the server could not find the requested resource
DEBUG Still waiting for the Kubernetes API: Get https://api.envisage.clientcenter.local:6443
DEBUG Still waiting for the Kubernetes API: Get https://api.envisage.clientcenter.local:6443
INFO API v1.16.2 up
INFO Waiting up to 30m0s for bootstrapping to complete...
```

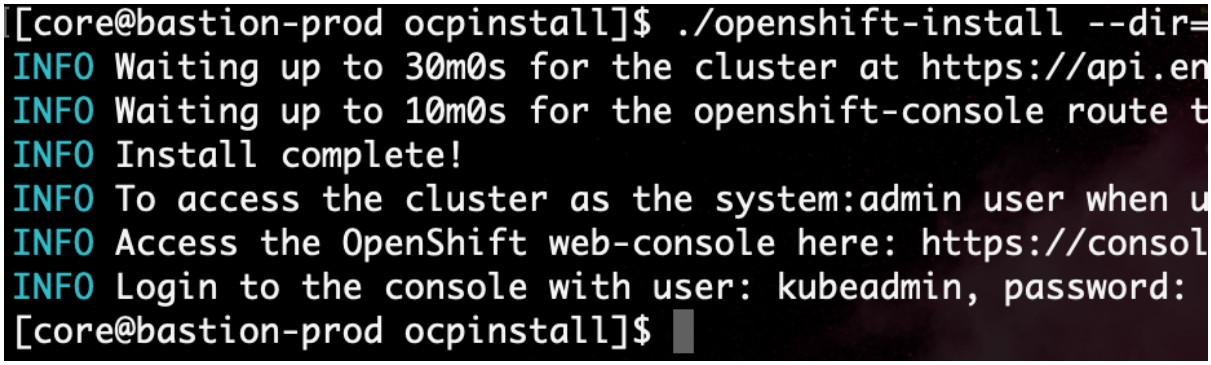
If you want to see more details about installers progress, you can start a tail
tail -f openshift_install.log

cd install_ocp4/vsphere

terraform apply -auto-approve -var 'bootstrap_complete=true'
cd install_ocp4

```
vsphere_resource_pool.resource_pool: Refreshing state... (ID: resgroup-787)
vsphere_virtual_machine.vm: Refreshing state... (ID: 42296b34-fd02-9cbd-9ebf-c81c7dec5fbe)
vsphere_virtual_machine.vm[1]: Refreshing state... (ID: 422903b3-2b94-7bb2-3c3a-ff804b055a6b)
vsphere_virtual_machine.vm[0]: Refreshing state... (ID: 422910fb-e10d-abb0-93f3-2eca247c0a8d)
vsphere_virtual_machine.vm[2]: Refreshing state... (ID: 4229c635-4a40-f4e2-cb27-a759682db6d1)
vsphere_virtual_machine.vm[0]: Refreshing state... (ID: 422919d6-cb13-bd82-361f-ec2c7bd1c642)
vsphere_virtual_machine.vm[1]: Refreshing state... (ID: 42294e9d-84d5-56e8-1b73-d80bf9c5d6f8)
vsphere_virtual_machine.vm[2]: Refreshing state... (ID: 42290a2d-6c83-398f-ce53-732a12f2057e)
vsphere_virtual_machine.vm[3]: Refreshing state... (ID: 42293da0-4ff9-1b02-92d3-2c500e53f788)
vsphere_virtual_machine.vm[4]: Refreshing state... (ID: 422946d0-227b-8858-4c9a-71dfc568f4af)
module.bootstrap.vsphere_virtual_machine.vm: Destroying... (ID: 42296b34-fd02-9cbd-9ebf-c81c7de
module.bootstrap.vsphere_virtual_machine.vm: Destruction complete after 10s

Apply complete! Resources: 0 added, 0 changed, 1 destroyed.
[core@Bastionhost vsphere]$
```

./openshift-install --dir=. wait-for install-complete


[core@bastion-prod ocpinstall]$ ./openshift-install --dir=
INFO Waiting up to 30m0s for the cluster at https://api.er
INFO Waiting up to 10m0s for the openshift-console route t
INFO Install complete!
INFO To access the cluster as the system:admin user when u
INFO Access the OpenShift web-console here: https://consol
INFO Login to the console with user: kubeadmin, password:
[core@bastion-prod ocpinstall]$