Dans cette mise en pratique, vous allez mettre en place un cluster Kubernetes sur le cloud provider DigitalOcean à l'aide de *kubeadm*.

Attention:

DigitalOcean, un cloud provider très connu et simple à utiliser. Cependant, comme pour l'ensemble des cloud providers (Google Compute Engine, Amazon AWS, Packet, Rackspace, …) l'instantiation de VMs est payante (peu cher pour un test de quelques heures cependant) et nécessite d'avoir un compte utilisateur provisionné avec quelques euros. Si vous ne souhaitez pas réaliser la manipulation, n'hésitez pas à suivre cet exercice sans l'appliquer.

Quelques prérequis

Installation de Kubectl

Le binaire kubectl est l'outil indispensable pour communiquer avec un cluster Kubernetes depuis la ligne de commande. Son installation est très bien documentée dans la documentation officielle que vous pouvez retrouver via le lien suivant: https://kubernetes.io/docs/tasks/tools/install-kubectl/

En fonction de votre environnement, vous trouverez les différentes options qui vous permettront d'installer kubectl

• si vous êtes sur macOS:

```
$ curl -L0 https://storage.googleapis.com/kubernetes-
release/release/v1.14.0/bin/darwin/amd64/kubectl
$ chmod +x ./kubectl
$ sudo mv ./kubectl /usr/local/bin/kubectl
```

• si vous êtes sur Linux

```
$ curl -L0 https://storage.googleapis.com/kubernetes-
release/release/v1.14.0/bin/linux/amd64/kubectl
$ chmod +x ./kubectl
$ sudo mv ./kubectl /usr/local/bin/kubectl
```

si vous êtes sur Windows

\$ curl -L0 https://storage.googleapis.com/kubernetesrelease/release/v1.14.0/bin/windows/amd64/kubectl.exe

note: si vous n'avez pas l'utilitaire curl vous pouvez télécharger kubectl v1.14.0 depuis ce https://storage.googleapis.com/kubernetesrelease/release/v1.14.0/bin/windows/amd64/kubectl.exe.

Afin d'avoir les utilitaires comme curl, je vous conseille d'utiliser Git for Windows (https://gitforwindows.org), vous aurez alors Git Bash, un shell très proche de celui que l'on trouve dans un environnement Linux.

Il vous faudra ensuite mettre kubectl.exe dans le PATH.

Création d'un compte sur DigitalOcean

Créez un compte depuis l'interface de DigitalOcean.

Depuis le menu Security suivez la procédure pour ajouter une clé ssh.

← → C	IOcean, LLC [US https://cloud.	digitalocean 🔍 🛧 🤷 🛼 🕅 🖉 🎭 🚳 🤷	♥ ❷ ◙ ₽ ⊠ Ø 0 ∎ 5 0 0 •
		> New SSH key Paste a copy of your public key in the space below. It should end in "pub". Learn more: This does not add an SSH key to your existing Droplets. To do so, follow the instructions here.	
		SSH key content *	
		Name	
		Add SSH Key	

Celle-ci sera utile dans la suite afin de permettre une connection ssh sans mot de passe sur les machines qui seront provisionnées.

Optionnel:

depuis le menu *API* générez un TOKEN, celui-ci vous sera utile dans la suite si vous souhaitez gérer les éléments d'infrastructure depuis la ligne de commande

\leftarrow \rightarrow C $$ DigitalC	cean, LLC [US https://cloud	I.digitalocean 으	☆ 🐴 🔚 📎 🚴	🔺 🧕 👍	🔍 🖉 📴 🛽	2 🗹 🔇 🛈 📙 🕷) 🎲 🤇
				×			
		New personal	access token				
		Token name					
		Enter token name		*			
		Soloct scopes					
		Read (default)	Write (optional)				

Provisionning

L'étape de provisionning de l'infrastructure consiste en la création des machines virtuelles qui seront utilisées pour le déploiement du cluster Kubernetes.

Il y a différentes façon de réaliser ce provisionning:

- depuis l'interface web de DigitalOcean
- en utilisant l'utilitaire doctl en ligne de commande
- en utilisant Terraform, un excellent produit de Hashicorp permettant une approche IaC (Infrastructure As Code)

Nous utiliserons ici l'interface web mais n'hésitez pas à passer par une autre méthode si vous le souhaitez.

Cliquez sur le bouton Create en haut à droite et sélectionnez Droplets



Sélectionnez l'image Ubuntu 18.04, et laissez les options par défaut dans la section Plan.

PROJECTS ^	Create Drople						
		ets					
+ New Project	Choose an image						
AAMAGE A	Distributions Container distributions Marketolace Custom images						
Proplets							
ubernetes				0			
olumes		E		Co			
atabases 💷	Obuntu		redola				
paces	18.04 x64 🗸 🗸	Select version V	Select version V S	elect version V	Select version V		
nages							
letworking							
fonitoring	Choose a plan						
PI	STARTER			PERFORMANC	E		
ISCOVER ^	Standar	d	General Purpose	NEW	CPU Optimized	d	
farketplace	^						
CCOUNT ^	Standard virtual machines with and dev/test environments.	a mix of memory and compute r	resources. Best for small proje	cts that can handle variable	levels of CPU performance, lil	ke blogs, web apps	
rofile							
		\$80/~~	\$160/mo	\$240/mo	\$320/mo	\$ 480/mc	
illing	\$ 40 /mo	+00/110					
iilling iecurity	\$40/mo \$0.060/hour	\$0.119/hour	\$0.238/hour	\$0.357/hour	\$0.476/hour	\$0.714/hour	

Choisissez ensuite un datacenter proche de chez vous, et sélectionnez la clé ssh que vous avez définie plus haut, cette clé sera copiée sur les VMs lors de leur création. Indiquez que vous souhaitez créer 3 VMs et donnez leur les noms *node-01*, *node-02*, *node-03*.

Q Search by	Droplet name or IP (Cmd+B)	Create V 💿 🤔 USAGE \$39.85
PROJECTS ^		
📀 Luc Juggery	New York San Francisco Amsterdam Singapore London	Frankfurt
+ New Project	1 2 3 1 2 2 3 1 1	1
MANAGE ^		
Droplets	Toronto Bangalore	
Kubernetes 💶	1 1	
Volumes		
Databases 🛄		
Spaces	Select additional options 👔	
Images	Private networking IPv6 User data Monitoring	
Networking		
Monitoring API	Add your SSH keys 1	
	New SEL Ver	
DISCOVER ^	udining udining	
Marketplace		
	Finalize and create	
	How many Droplets? Choose a hostname	
Prome	Deploy multiple Droplets with the same configuration. Give your Droplets an identifying name you will remember them by. Your D	Proplet name can only contain
Security	alphanumeric characters, dashes, and periods.	
Referrals	- 3 Droplets + node+01	
	50.ebc	

Cliquez ensuite sur *Create*, les machines seront disponibles après quelques dizaines de secondes.

ລ	Q. Search by Droplet name or IP (Cmd+B)			Create → (2) (2) (2) (2) (2) (2) (2) (2) (2) (2)
 Luc Juggery + New Project 	Droplets		Search by Droplet name	
	Name	IP Address	Created A Tags	
MANAGE ^	(a) 8 GB / 160 GB Disk / LONI - Ubuntu 18.04 x64	178.62.69.201	Let's get to work!	More 🗸
Kubernetes	node-02 so / 450 GP Disk / LONI, Liberto 19.04 v64	178.62.58.44	Go change the world!	More 🗸
Volumes				
Databases	8 GB / 160 GB Disk / LONI - Ubuntu 18.04 x64	206.189.17.156	Happy coding!	More 🗸
Images				
Networking				
Monitoring API				
Marketplace				
Profile				
Billing				
Security				
Referrals				

Note: chaque VM de ce type coute 40\$ / mois. Pour un cluster de 3 machines que l'on utilise durant 4 heures, cela reviendra donc à mois d'1\$. Attention cependant à bien supprimer vos

VMs lorsque vous ne les utilisez plus.

Configuration

L'étape de configuration consiste à installer les logiciels nécessaires sur l'infrastructure provisionnée lors de l'étape précédente.

Il y a différentes façon de réaliser cette configuration:

- en se connectant manuellement en ssh sur chaque machine
- en utilisant un utilitaire de configuration, comme Ansible, Chef, Puppet

Nous lancerons ici des commandes via ssh mais n'hésitez pas à passer par une autre méthode si vous le souhaitez.

Installation des prérequis

Sur chaque machine, nous allons installer les éléments suivants:

- un runtime de container (nous utiliserons Docker)
- le binaire kubeadm pour la création du cluster
- le binaire kubelet pour la supervision des containers

Pour chaque VM, connectez vous en ssh et effectuez les opérations suivantes:

• Connection en ssh

Assurez-vous que la clé utilisée par les nodes (celle sélectionnée durant l'étape de création) est présente dans la liste de vos identités. Si ce n'est pas le cas, ajoutez la via la commande ssh-add PATH_T0_KEY.

\$ ssh root@NODE_IP

• Installation du daemon Docker avec la commande suivante:

root@node-X:~# curl https://get.docker.com | sh

Installation de kubelet et kubeadm

```
root@node-X:~# curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg |
apt-key add -
root@node-X:~# cat <<EOF >/etc/apt/sources.list.d/kubernetes.list
deb https://apt.kubernetes.io/ kubernetes-xenial main
EOF
root@node-X:~# apt-get update
root@node-X:~# apt-get install -y kubelet kubeadm
```

Initialisation du cluster

Sur le node-01, lancez la commande suivante:

```
root@node-01:~# kubeadm init
```

Ajout de nodes

La mise en place de l'ensemble des composants du master prendra quelques minutes. A la fin vous obtiendrez la commande à lancez depuis les autres VMs pour les ajouter au cluster. Copiez cette commande et lancez la depuis *node-02* et *node-03*.

Exemple de commande lancée depuis le node-02:

Récupération du context

Afin de pouvoir dialoguer avec le cluster que vous venez de mettre en place, vi le binaire *kubectl* que vous avez installé sur votre machine locale, il est nécessaire de récupérer le fichier de configuration du cluster. Utilisez pour cela les commandes suivantes depuis votre machine locale:

```
$ scp root@IP_NODE_01:/etc/kubernetes/admin.conf do-kube-config
$ sudo chown $(id -u):$(id -g) do-kube-config
$ export KUBECONFIG=do-kube-config
```

Listez alors les nodes du cluster, ils apparaitront avec le status *NotReady*.

get nodes			
STATUS	ROLES	AGE	VERSION
NotReady	master	5m16s	v1.14.1
NotReady	<none></none>	3m	v1.14.1
NotReady	<none></none>	10s	v1.14.1
	get nodes STATUS NotReady NotReady NotReady	get nodes STATUS ROLES NotReady master NotReady <none> NotReady <none></none></none>	get nodes STATUS ROLES AGE NotReady master 5m16s NotReady <none> 3m NotReady <none> 10s</none></none>

Installation d'un addons pour le networking entre les Pods

La commande suivante permet d'installer les composants nécessaires pour la communication entre les Pods qui seront déployés sur le cluster.

\$ kubectl apply -f "https://cloud.weave.works/k8s/net?k8s-version=\$(kubectl
version | base64 | tr -d '\n')"

Note: il y a plusieurs solutions de networking qui peuvent être utilisées, la solution envisagées ici est Weave Net. L'article suivant donne une bonne comparaison des solutions les plus utilisées: objectif-libre.com/fr/blog/2018/07/05/comparatif-solutions-reseaux-kubernetes/

Vérification de l'état de santé des nodes

Maintenant que la solution de netorking a été installée, les nodes sont dans l'état Ready.

<pre>\$ kubectl</pre>	. get node	es		
NAME	STATUS	ROLES	AGE	VERSION
node <mark>-01</mark>	Ready	master	6m1s	v1.14.1
node <mark>-02</mark>	Ready	<none></none>	<mark>3m</mark> 45s	v1.14.1
node <mark>-03</mark>	Ready	<none></none>	55s	v1.14.1

Le cluster est prêt à être utilisé.