

4Kube

Par LARIBIERE Bruno - 298261

Installation

Voir chapitre **flux**

Vérification

Après le déploiement automatique avec flux la node sur laquelle est déployée le front devrait exposer le site sur le port 30000.

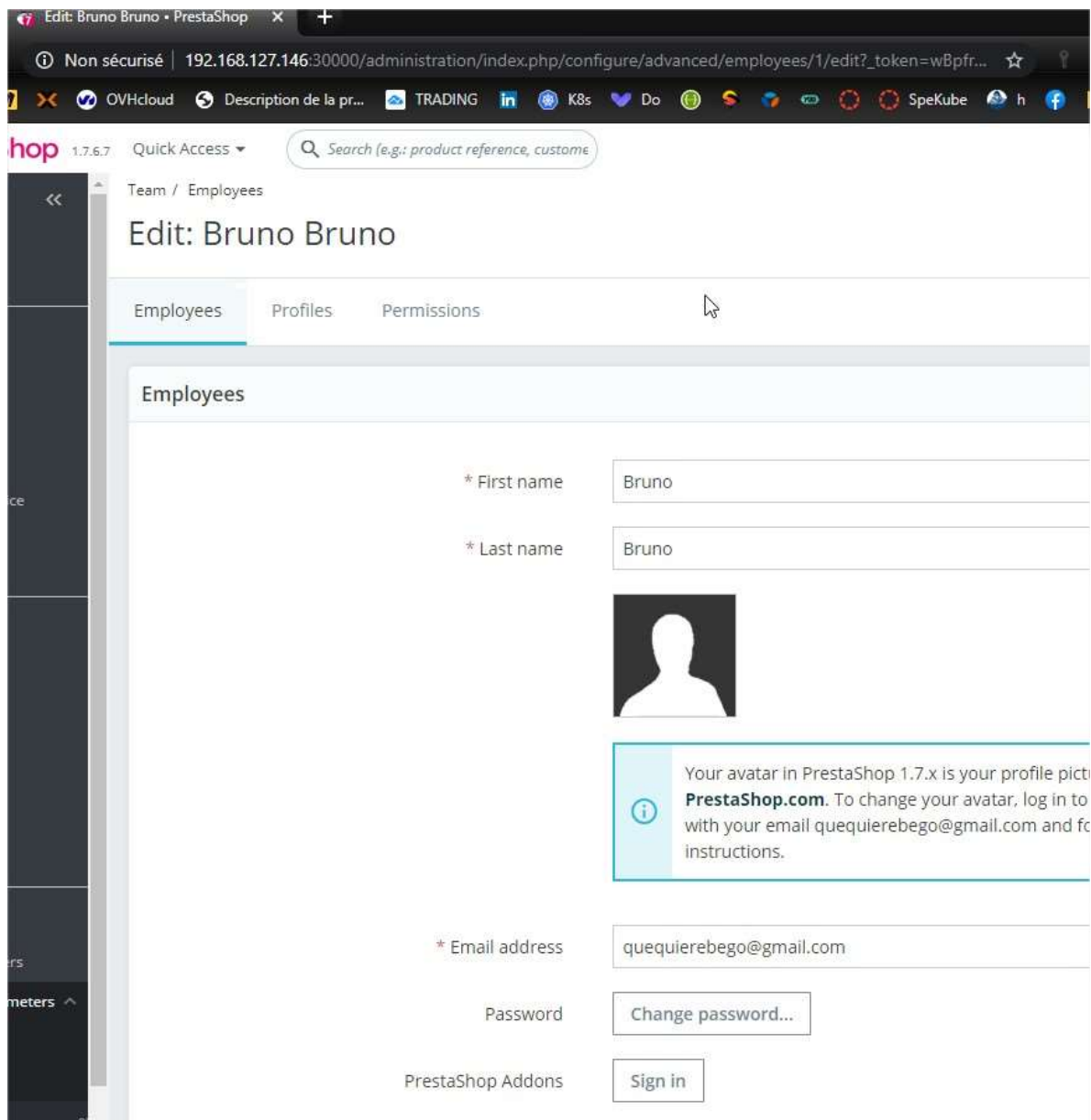
Dans un premier temps, les logs de prestashop doivent nous confirmer la bonne prise en compte des credentials:

```
ami INFO Initializing mysql-client
ami INFO mysql-client successfully initialized
ami INFO Initializing prestashop
restas INFO Configuring webserver...
restas INFO Configuring PHP settings...
mysql-c INFO Trying to connect to MySQL server
mysql-c INFO Found MySQL server listening at mariasql-svc:3306
mysql-c INFO MySQL server listening and working at mariasql-svc:3306
restas INFO Configuring PrestaShop...
restas INFO #####
restas INFO Installation parameters for prestashop:
restas INFO   First Name: bruno
restas INFO   Last Name: bruno
restas INFO   Email: quequierebego@gmail.com
restas INFO   Password: *****
restas INFO   Shop Name: PrestaShop
restas INFO   Admin URL: http://[192.168.127.146:30000]/administration
restas INFO   (Passwords are not shown for security reasons)
restas INFO #####
restas INFO
ami INFO prestashop successfully initialized
```

Nous pouvons donc ensuite aller sur l'ip de notre node (192.168.127.146:30000):

```
C:\Users\quequ\source\repos\Supinfo4k>kubectl get nodes -o wide
NAME      STATUS    ROLES    AGE   VERSION   INTERNAL-IP   EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION   CONTAINER-RUNTIME
kube4     Ready     master   8d    v1.16.3   192.168.127.146   <none>         Ubuntu 20.04 LTS     5.4.0-40-generic   docker://19.3.8
```

Et pouvons constater que prestashop est bien en ligne avec nos credentials



Nous confirmons donc que le déploiement s'est bien effectué avec succès.

Détails manifestes

Les manifestes se trouvent dans `./releases/prestans`

1-config.yaml

Configmap contenant des données non confidentielles comme le nom et prénom à utiliser pour le profil administrateur prestashop

1-secrets.yaml

Contient l'ensemble des données privées, comme les passwords prestashop ou bien ceux de la database

2-persistence.yaml

Décrit la méthode de stockage sur le host afin de pouvoir stocker ensuite les données de la database

3-mariadb.yaml

Déploiement d'une DB mariadb. Nous utilisons les secrets en tant que variable d'environnements

4-mariasql-svc.yaml

Expose la mariadb uniquement sur le cluster afin qu'il ne soit pas accessible depuis l'extérieur

4-prestashop.yaml

Déploie prestashop. Le PRESTASHOP_HOST permet que le site fonctionne correctement avec une ip interne.

5-prestashop-svc.yaml

Expose prestashop sur le nœud.

FluxCD

FluxCD va s'occuper du déploiement de nos manifestes. Pour cela nous admettons par avance que le client fluxctl a été installé sur la machine cliente.

Installation de flux

```
kubectl create ns flux

helm upgrade -i flux fluxcd/flux --set
git.url=git@github.com:quequiere/4kubetemp --set git.defaultRef=master --
namespace flux
```

Ceci nous permet d'installation flux dans le namespace flux, et de lui dire de rester à l'écoute du repository github **quequiere/4kubetemp**

Synchronisation git

A cette étape nous devons rajouter la clef RSA qui permettra à flux d'avoir assez d'accès sur notre repository.

Plusieurs choix pour obtenir la clef

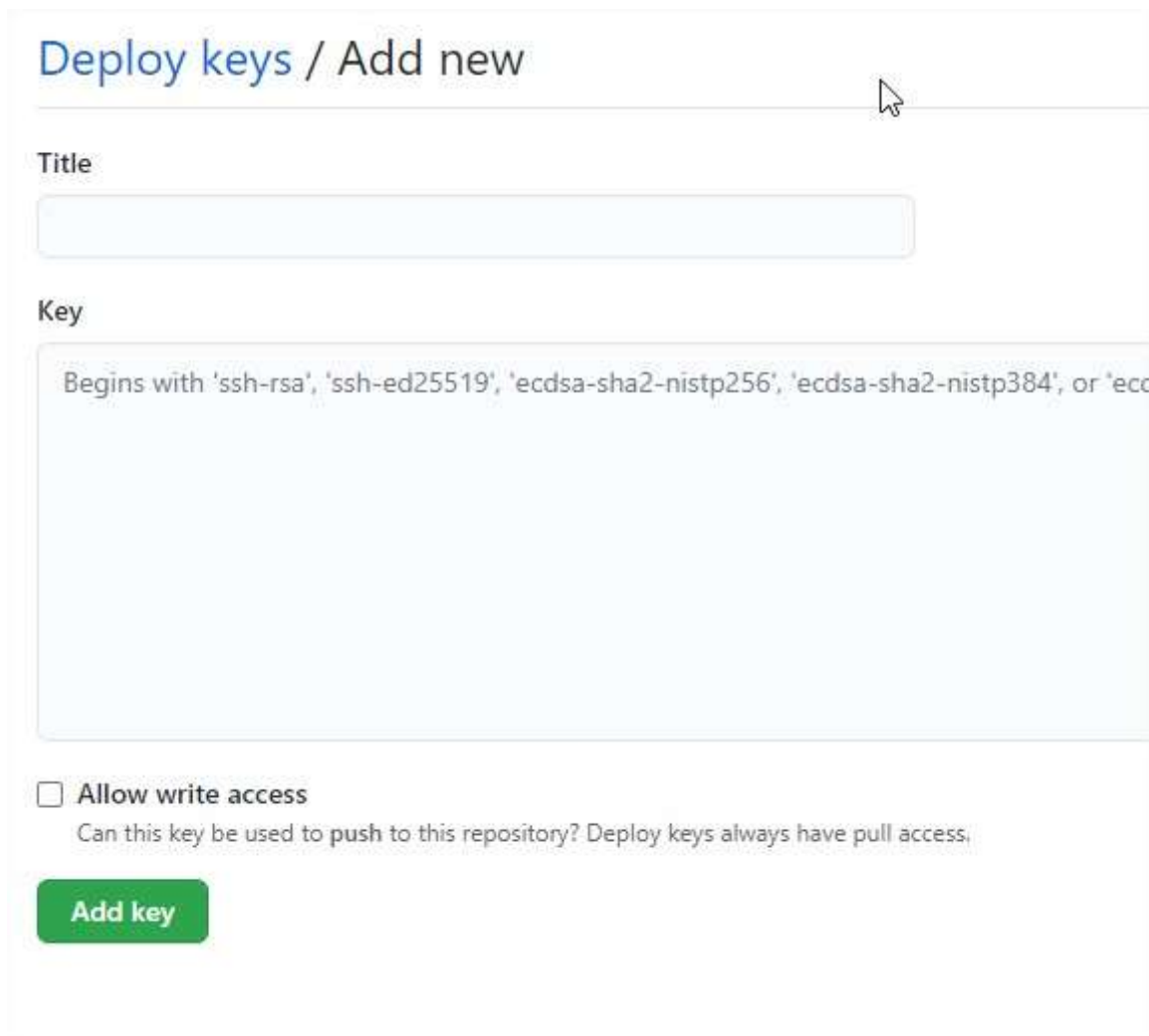
Regarder les logs à l'installation de flux

```
>>kubectl -n flux logs -f deployment/flux
recreated, changed to --git-verify-signatures-mode, use that instead
-main.go:259 version=1.20.0
-main.go:412 msg="using kube config: \"/root/.kube/config\" to connect to the cluster"
-main.go:492 component=cluster identity=/var/fluxd/keygen/identity
-main.go:493 component=cluster identity.pub="ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGCovfxz2pd2yXGff/5zQD1Bj7J13ClT3RZ251Qiv6bLc2zrCVNskfivPDuWjZdG1GEJ8
9VD7rOyXIw8skovSKuizOppT7K/NF6YvmUDajjWjkt2gAssdwyFX9/KYr4XBLb5Vg+IN8/LBEVGu+9ZgkCjXtBmZs7KubZDtSkgoDo8v2SztTtEluJ+I8X04Wnzuo6BERo3v1LcUec/+kHyTL5jn2u
HRUp6lKByczUQ1vVbYETs6PoZKTztGa2GsQ10oT870QFn7NDT3Jztui3JoeymioT+kvZJ+X4Nt/1Bt5DawHA5V1Po9KzQziyBSODbXFYeumVnouZvr/jnLuHFX1XEEnGFWJHxxEIwD6ji/Ptq/4fe
p1WeFlz2kRu3VU+8910aH6Xe7c- root@flux-6b5fcb48f6-wdmz8"
-main.go:498 host=https://10.96.0.1:443 version=kubernetes-v1.16.13
-main.go:510 kubectl=/usr/local/bin/kubectl
-main.go:527 ping=true
-main.go:666 url=ssh://git@github.com:quequiere/4kubetemp user="Weave Flux" email=support@weave.works signing-key= verify-signatures-mode=none sync-ta
istry-disable-scanning=false notes-ref=flux set-author=false git-secret=false sops=false
-main.go:772 upstream="no upstream URL given"
-main.go:795 addr=:3030
-loop.go:108 component=sync-loop err="loading last-synced resources: git repo not ready: git repo has not been cloned yet"
-images.go:17 component=sync-loop msg="polling for new images for automated workloads"
-images.go:27 component=sync-loop msg="no automated workloads"
-checkpoint.go:24 component=checkpoint msg="up to date" latest=1.20.0
-warming.go:198 component-warmer info="refreshing image" image=quay.io/coreos/flannel tag_count=87 to_update=87 of_which_refresh=0 of_which_missing=87
```

Utiliser la commande suivante

```
fluxctl identity --k8s-fwd-ns flux
```

Une fois la clef obtenue on l'ajoute dans les accès de notre repository:



Deploy keys / Add new

Title

Key

Begins with 'ssh-rsa', 'ssh-ed25519', 'ecdsa-sha2-nistp256', 'ecdsa-sha2-nistp384', or 'ecdsa-sha2-nistp256'

☐ Allow write access
Can this key be used to push to this repository? Deploy keys always have pull access.

Add key

Il ne reste plus qu'à attendre la synchronisation. Il est possible de la forcer avec la commande suivante:

```
fluxctl sync --k8s-fwd-ns flux
```

Pour suivre le bon déroulement de la synchronisation nous pouvons regarder les logs de cette manière:

```
kubectl -n flux logs -f deployment/flux
```

Fin du déploiement

Après la synchronisation l'ensemble des fichiers vont être poussés depuis **./releases/prestans**

Le namespace sera aussi créé automatiquement vu qu'il est présent dans **./namespaces**

Bonus

Quelques difficulté d'intégrations on fait que je n'ai pas réussi à introduire mon package helm dans le déploiement automatique avec flux.

Cependant le package helm est bien fonctionnel et peut être trouvé dans **./BONUS.HELM**

Pour l'installation utiliser la commande

```
helm install 4kube .
```

Pour la suppression

```
helm delete 4kube
```

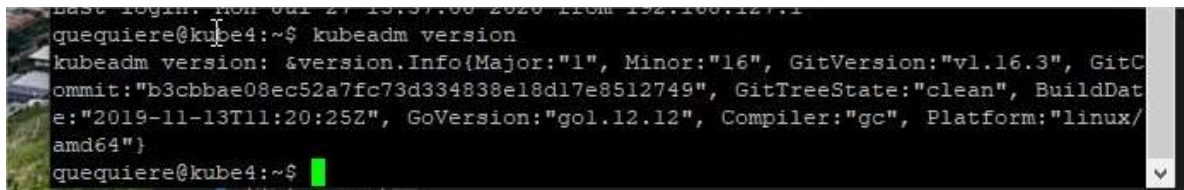
Rien de particulier à signaler si ce n'est que bien entendu l'ensemble des valeurs est défini dans le values.yaml

Mise à jour du cluster

Dans mon cas mon cluster a été fait mis en place sur un nœud unique. Mes screens se baseront donc là dessus.

Vérification de la version

Nous vérifions que la versions de kubeadm est bien en 1.16

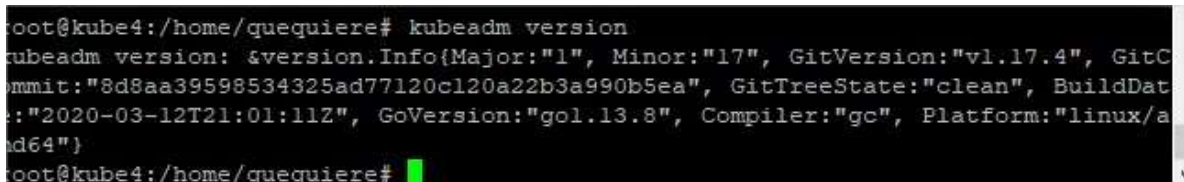
A terminal window with a dark background. The prompt is 'quequiere@kub4:~\$'. The command 'kubeadm version' has been executed. The output shows: 'kubeadm version: &version.Info{Major:"1", Minor:"16", GitVersion:"v1.16.3", GitCommit:"b3cbbae08ec52a7fc73d334838e18dl7e8512749", GitTreeState:"clean", BuildDate:"2019-11-13T11:20:25Z", GoVersion:"go1.12.12", Compiler:"gc", Platform:"linux/amd64"}'. The prompt is now 'quequiere@kub4:~\$'.

Installation de kubeadm 1.17.4

Nous installons la nouvelle version de kubeadm avec la commande suivante:

```
apt-mark unhold kubeadm && \
apt-get update && apt-get install -y kubeadm=1.17.4-00 && \
apt-mark hold kubeadm
```

Et nous confirmons que la version a été correctement installée (1.17.4)

A terminal window with a dark background. The prompt is 'root@kub4:/home/quequiere#'. The command 'kubeadm version' has been executed. The output shows: 'kubeadm version: &version.Info{Major:"1", Minor:"17", GitVersion:"v1.17.4", GitCommit:"8d8aa39598534325ad77120cl20a22b3a990b5ea", GitTreeState:"clean", BuildDate:"2020-03-12T21:01:11Z", GoVersion:"go1.13.8", Compiler:"gc", Platform:"linux/amd64"}'. The prompt is now 'root@kub4:/home/quequiere#'.

Update du cluster

Drainage de nœud que nous voulons mettre à jour afin d'éviter l'interruption de service.

(Non exécuté chez moi car un seul noeud)

```
kubect1 drain kube4 --ignore-daemonsets
```

On prépare la procédure de migration:

```
sudo kubeadm upgrade plan
```

```
[upgrade/versions] Latest stable version: v1.17.9
[upgrade/versions] Latest version in the v1.16 series: v1.16.13

Components that must be upgraded manually after you have upgraded the control plane with 'kubeadm upgrade apply':
COMPONENT      CURRENT      AVAILABLE
Kubelet        1 x v1.16.3  v1.17.9

Upgrade to the latest stable version:

COMPONENT      CURRENT      AVAILABLE
API Server     v1.16.13    v1.17.9
Controller Manager v1.16.13    v1.17.9
```

Tout semble ok et prêt pour la migration que nous confirmons avec la commande suivante:

```
sudo kubeadm upgrade apply v1.17.4
```

```
[bootstrap-token] configured RBAC rules to allow certificate rotation for all node client certificates in the cluster
[addons]: Migrating CoreDNS Corefile
[addons] Applied essential addon: CoreDNS
[addons] Applied essential addon: kube-proxy

[upgrade/successful] SUCCESS! Your cluster was upgraded to "v1.17.4". Enjoy!

[upgrade/kubelet] Now that your control plane is upgraded, please proceed with u
```

Ce message me confirme que la mise à jour s'est déroulée avec succès.