

Enzo Allemanno
Yohann Paulus
Luca Zacheo

19/05/2022

Rémi Poulard
Marcel Graf

Cloud Computing

Rapport laboratoire 6: Kubernetes

Task 1 : DEPLOY THE APPLICATION ON A LOCAL TEST CLUSTER

Description des objets :

Services :

Name: api-svc Namespace: default Labels: component=api Annotations: <none> Selector: app=todo,component=api Type: ClusterIP IP Family Policy: SingleStack IP Families: IPv4 IP: 10.107.204.100 IPs: 10.107.204.100 Port: api 8081/TCP TargetPort: 8081/TCP Endpoints: 172.17.0.7:8081 Session Affinity: None Events: <none>	Name: kubernetes Namespace: default Labels: component=apiserver provider=kubernetes Annotations: <none> Selector: <none> Type: ClusterIP IP Family Policy: SingleStack IP Families: IPv4 IP: 10.96.0.1 IPs: 10.96.0.1 Port: https 443/TCP TargetPort: 8443/TCP Endpoints: 192.168.59.100:8443 Session Affinity: None Events: <none>
Name: redis-svc Namespace: default Labels: component=redis Annotations: <none> Selector: app=todo,component=redis Type: ClusterIP IP Family Policy: SingleStack IP Families: IPv4 IP: 10.105.158.116 IPs: 10.105.158.116 Port: redis 6379/TCP TargetPort: 6379/TCP Endpoints: 172.17.0.2:6379 Session Affinity: None Events: <none>	

Pods

```

Name:          api
Namespace:     default
Priority:       0
Node:          minikube/192.168.59.100
Start Time:    Tue, 17 May 2022 11:08:07 +0200
Labels:        app=todo
               component=api
Annotations:   <none>
Status:        Running
IP:            172.17.0.7
IPs:
  IP: 172.17.0.7
Containers:
  api:
    Container ID:  docker://029d69d5745c381d9d7d301999d5a4573bd677a1b4df074abe2b9309be9dae82
    Image:         icclabcna/ccp2-k8s-todo-api
    Image ID:      docker-pullable://icclabcna/ccp2-k8s-todo-
api@sha256:13cb50bc9e93fdf10b4608f04f2966e274470f00c0c9f60815ec8fc987cd6e03
    Port:         8081/TCP
    Host Port:     0/TCP
    State:         Running
      Started:     Tue, 17 May 2022 11:08:31 +0200
    Ready:         True
    Restart Count: 0
    Environment:
      REDIS_ENDPOINT: redis-svc
      REDIS_PWD:      ccp2
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-m6pf2 (ro)
Conditions:
  Type            Status
  Initialized      True
  Ready            True
  ContainersReady  True
  PodScheduled     True
Volumes:
  kube-api-access-m6pf2:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:    kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:      true
QoS Class:         BestEffort
Node-Selectors:    <none>
Tolerations:       node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                   node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

```

```

Name:      frontend
Namespace: default
Priority:   0
Node:      minikube/192.168.59.100
Start Time: Tue, 17 May 2022 10:56:20 +0200
Labels:    app=todo
           component=frontend
Annotations: <none>
Status:     Running
IP:         172.17.0.4
IPs:
  IP: 172.17.0.4
Containers:
  api:
    Container ID:  docker://c17555213d578b8da6081ba38b6e402509b48fab128ea2ab4c2c8fa59db047c9
    Image:         icclabcna/ccp2-k8s-todo-frontend
    Image ID:      docker-pullable://icclabcna/ccp2-k8s-todo-frontend@sha256:5892b8f75a4dd3aa9d9cf527f8796a7638dba574ea8e6beef49360a3c67bbb44
    Port:         8080/TCP
    Host Port:    0/TCP
    State:        Running
      Started:    Tue, 17 May 2022 10:56:25 +0200
    Ready:        True
    Restart Count: 0
    Environment:
      API_ENDPOINT_URL: http://api-svc:8081
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-tksrv (ro)
Conditions:
  Type            Status
  Initialized      True
  Ready            True
  ContainersReady  True
  PodScheduled     True
Volumes:
  kube-api-access-tksrv:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:    kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:      true
QoS Class:         BestEffort
Node-Selectors:    <none>
Tolerations:       node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                   node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

```

```

Name:      redis
Namespace: default
Priority:   0
Node:      minikube/192.168.59.100
Start Time: Tue, 17 May 2022 10:40:42 +0200
Labels:    app=todo
           component=redis
Annotations: <none>
Status:     Running
IP:         172.17.0.2
IPs:
  IP: 172.17.0.2
Containers:
  redis:
    Container ID: docker://6b05133f6abb29b51b9d3dcd464413b372b3dfce1d34ed9b31e594c658c1c9f7
    Image:        redis
    Image ID:     docker-pullable://redis@sha256:ad0705f2e2344c4b642449e658ef4669753d6eb70228d46267685045bf932303
    Port:         6379/TCP
    Host Port:    0/TCP
    Args:
      redis-server
      --requirepass ccp2
      --appendonly yes
    State:        Running
      Started:    Tue, 17 May 2022 10:40:45 +0200
    Ready:        True
    Restart Count: 0
    Environment:  <none>
    Mounts:
      /var/run/secrets/kubernetes.io/serviceaccount from kube-api-access-vxhzz (ro)
Conditions:
  Type              Status
  Initialized        True
  Ready              True
  ContainersReady    True
  PodScheduled       True
Volumes:
  kube-api-access-vxhzz:
    Type:          Projected (a volume that contains injected data from multiple sources)
    TokenExpirationSeconds: 3607
    ConfigMapName:    kube-root-ca.crt
    ConfigMapOptional: <nil>
    DownwardAPI:      true
QoS Class:          BestEffort
Node-Selectors:      <none>
Tolerations:         node.kubernetes.io/not-ready:NoExecute op=Exists for 300s
                     node.kubernetes.io/unreachable:NoExecute op=Exists for 300s

```

Tache 2 : DEPLOY THE APPLICATION IN KUBERNETES ENGINE

Les pods du frontend, de l'api et de redis ne changent pas.

Les services de l'api, de redis et de kubernetes ne changent pas.

Le service du frontend est créé avec le type « LoadBalancer » et non pas « ClusterIP »

Service

```
Name: frontend-svc
Namespace: default
Labels: component=frontend
Annotations: cloud.google.com/neg: {"ingress":true}
Selector: app=todo,component=frontend
Type: LoadBalancer
IP Family Policy: SingleStack
IP Families: IPv4
IP: 10.68.8.54
IPs: 10.68.8.54
LoadBalancer Ingress: 34.116.217.225
Port: frontend 80/TCP
TargetPort: 8080/TCP
NodePort: frontend 30788/TCP
Endpoints: 10.64.1.7:8080
Session Affinity: None
External Traffic Policy: Cluster
Events: <none>
```

Détail du cluster depuis la console de GKE

<div>  gke-cluster-1 </div>		
DÉTAILS	NŒUDS	STOCKAGE
JOURNAUX		
Paramètres de base du cluster		
Nom	gke-cluster-1	
Type d'emplacement	Zonal	
Zone du plan de contrôle	europe-central2-c	
Zones de nœuds par défaut 	europe-central2-c	
Version disponible	Version standard	 MISE À NIVEAU DISPONIBLE
Version	1.21.11-gke.900	
Taille totale	4	
Point de terminaison	34.116.195.36 Afficher le certificat du cluster	

Tache 3 : ADD AND EXERCISE RESILIENCE

Après avoir kill l'ensemble des pods et recréé leur version cette fois-ci au format « deployment » on obtient l'output suivant avec la commande `kubectl get all`.

NAME	READY	STATUS	RESTARTS	AGE
pod/api-deploy-86d8969586-44j6g	1/1	Running	0	5h2m
pod/api-deploy-86d8969586-wxg6m	1/1	Running	0	5h2m
pod/frontend-deploy-b5669674f-t68nr	1/1	Running	0	4h53m
pod/frontend-deploy-b5669674f-zwnt2	1/1	Running	0	5h2m
pod/redis-deploy-6fbcc669d9-xttf9	1/1	Running	0	5h2m

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
service/api-svc	ClusterIP	10.68.12.166	<none>	8081/TCP	5h37m
service/frontend-svc	LoadBalancer	10.68.8.54	34.116.217.225	80:30788/TCP	5h49m
service/kubernetes	ClusterIP	10.68.0.1	<none>	443/TCP	28h
service/redis-svc	ClusterIP	10.68.4.242	<none>	6379/TCP	5h38m

NAME	READY	UP-TO-DATE	AVAILABLE	AGE
deployment.apps/api-deploy	2/2	2	2	5h2m
deployment.apps/frontend-deploy	2/2	2	2	5h2m
deployment.apps/redis-deploy	1/1	1	1	5h2m

NAME	DESIRED	CURRENT	READY	AGE
replicaset.apps/api-deploy-86d8969586	2	2	2	5h2m
replicaset.apps/frontend-deploy-b5669674f	2	2	2	5h2m
replicaset.apps/redis-deploy-6fbcc669d9	1	1	1	5h2m

Tout semble fonctionner correctement. On peut notamment voir :

- Les 4 services créés au paravent
- Les 3 pods qui sont maintenant des « deployment »
- Le replicat des 3 pods
- Le nombre attendu de pods déployés

On peut également constater que les replica sets sont fonctionnels à l'aide de la manip suivante.

Premièrement, on regarde l'état des pods en continue avec `kubectl get pods --watch`

NAME	READY	STATUS	RESTARTS	AGE
api-deploy-86d8969586-44j6g	1/1	Running	0	5h16m
api-deploy-86d8969586-wxg6m	1/1	Running	0	5h16m
frontend-deploy-b5669674f-t68nr	1/1	Running	0	5h7m
frontend-deploy-b5669674f-zwnt2	1/1	Running	0	5h16m
redis-deploy-6fbcc669d9-xttf9	1/1	Running	0	5h16m

Ensuite, on kill un des pods du frontend avec

```
kubectl delete pod/frontend-deploy-b5669674f-t68nr
```

On peut remarquer que notre affichage en continue de la listes des pods affiche maintenant :

frontend-deploy-b5669674f-t68nr	1/1	Terminating	0	5h10m
frontend-deploy-b5669674f-sk2zp	0/1	Pending	0	0s
frontend-deploy-b5669674f-sk2zp	0/1	Pending	0	0s
frontend-deploy-b5669674f-sk2zp	0/1	ContainerCreating	0	0s
frontend-deploy-b5669674f-t68nr	0/1	Terminating	0	5h10m
frontend-deploy-b5669674f-sk2zp	1/1	Running	0	3s
frontend-deploy-b5669674f-t68nr	0/1	Terminating	0	5h11m
frontend-deploy-b5669674f-t68nr	0/1	Terminating	0	5h11m

On constate donc que le pods a bien été kill et qu'il a été relayé par un nouveau. Durant ce cours laps de temps, notre api était toujours disponible car c'est le deuxième pods qui a momentanément pris le relais.

Réponse aux questions

Why only use 1 instance for the Redis-Server ?

Dans cette infrastructure, nous ne voulons pas de duplica des données sur la DB, il n'est donc pas nécessaire de créer une deuxième instance pour Redis-Server.

What happens if you delete a Frontend or API Pod? How long does it take for the system to react?

Comme on peut le constater sur le screenshot au-dessus, la commande pour delete un pod commence par mettre le pod ciblé dans l'état Terminating. Une fois qu'un nouveau pod est créé et est en « attente » d'être utilisé, le pod ciblé est définitivement supprimé et le nouveau pod prend le relais. Cette opération est relativement rapide (~3sec) et serait quasiment invisible pour un utilisateur qui se trouverait sur l'application web.

What happens when you delete the Redis Pod?

Comme on peut s'y attendre un nouveau pod Redis est créé.

Cependant on note les choses suivantes :

- les données ont été perdues ! Puisque la DB se trouve sur le volume du pods et que le volume d'un pod n'est pas persistant, celle-ci est supprimée en même temps que le pod.
- L'application (api pod) n'est plus accessible car elle essaye toujours de contacter la DB qui vient d'être supprimée. Un simple redémarrage du pod api permet de mettre à jour la bonne adresse de la DB.

How can you change the number of instances temporarily to 3? Hint: look for scaling in the deployment documentation

Sur GK -> Dans les détails des Deployment -> Action -> Scale -> augmenter/diminuer le nombre de replicas

On peut aussi utiliser la commande

```
kubectl scal deployment/frontend-deploy --replicas={nb_de_replica}
```


What autoscaling features are available? Which metrics are used?

Les fonctionnalités pour l'autoscaling peuvent agir sur le nombre des pods. Les métriques à disposition pour gérer l'autoscaling sont l'utilisation du CPU et de la mémoire, ainsi que certaines metrics customisées.

How can you update a component? (see "Updating a Deployment" in the deployment documentation)

[Deployments | Kubernetes](#)

Avec la commande : `kubectl edit deployment/frontend-deploy`

Avec la commande :

`kubectl set image deployment/nginx-deployment nginx=nginx:1.16.1`

Avec l'invité graphique en passant par l'action : Rolling update

Description des deployments

```

Name:          api-deploy
Namespace:     default
CreationTimestamp: Wed, 18 May 2022 11:06:23 +0200
Labels:        app=todo
                component=api
Annotations:   deployment.kubernetes.io/revision: 1
Selector:      app=todo,component=api
Replicas:      2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:  RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=todo
           component=api
  Containers:
    api:
      Image:   icclabcna/ccp2-k8s-todo-api
      Port:    8081/TCP
      Host Port: 0/TCP
      Environment:
        REDIS_ENDPOINT: redis-svc
        REDIS_PWD:      ccp2
      Mounts:           <none>
      Volumes:          <none>
Conditions:
  Type           Status  Reason
  ----           -
  Available      True    MinimumReplicasAvailable
  Progressing    True    NewReplicaSetAvailable
OldReplicaSets: <none>
NewReplicaSet:  api-deploy-86d8969586 (2/2 replicas created)
Events:         <none>

```

```

Name:          frontend-deploy
Namespace:     default
CreationTimestamp: Wed, 18 May 2022 11:06:28 +0200
Labels:        app=todo
                component=frontend
Annotations:   deployment.kubernetes.io/revision: 1
Selector:      app=todo,component=frontend
Replicas:      2 desired | 2 updated | 2 total | 2 available | 0 unavailable
StrategyType:  RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=todo
           component=frontend
  Containers:
    frontend:
      Image:   icclabcna/ccp2-k8s-todo-frontend
      Port:    8080/TCP
      Host Port: 0/TCP
      Requests:
        cpu: 100m
      Environment:
        API_ENDPOINT_URL: http://api-svc:8081
      Mounts:           <none>
      Volumes:          <none>
Conditions:
  Type           Status  Reason
  ----           -
  Progressing    True    NewReplicaSetAvailable
  Available      True    MinimumReplicasAvailable
OldReplicaSets: <none>
NewReplicaSet:  frontend-deploy-b5669674f (2/2 replicas created)
Events:         <none>

```

```

Name:          redis-deploy
Namespace:     default
CreationTimestamp: Wed, 18 May 2022 11:06:12 +0200
Labels:        app=todo
                component=redis
Annotations:   deployment.kubernetes.io/revision: 1
Selector:      app=todo,component=redis
Replicas:      1 desired | 1 updated | 1 total | 1 available | 0 unavailable
StrategyType:  RollingUpdate
MinReadySeconds: 0
RollingUpdateStrategy: 25% max unavailable, 25% max surge
Pod Template:
  Labels:  app=todo
           component=redis
  Containers:
    redis:
      Image:      redis
      Port:       6379/TCP
      Host Port:  0/TCP
      Args:
        redis-server
        --requirepass ccp2
        --appendonly yes
      Environment: <none>
      Mounts:      <none>
      Volumes:     <none>
  Conditions:
    Type           Status  Reason
    ----           -
    Available      True    MinimumReplicasAvailable
    Progressing    True    NewReplicaSetAvailable
OldReplicaSets: <none>
NewReplicaSet:  redis-deploy-6fbcc669d9 (1/1 replicas created)
Events:         <none>

```

Descriptions des pods

La description des pods reste la même qu'à la tâche 2, à l'exception du paramètre `pod-template-hash` qui vient de se rajouter dans chacun des pods. Comme le décrit le nom du paramètre, celui-ci contient le hash du template décrit dans le deployment correspondant.

Tache 4 : DEPLOY ON IICT KUBERNETES CLUSTER

Les paramètres de l'infrastructure fonctionnent toujours correctement. Nous avons les mêmes résultats pour les test de fonctionnement des replica sets que pour l'étape 3.

La description des objets lors du déploiements dans kubernetes ne change pas. A l'exception des points suivants :

Deployments

Le namespace est maintenant égale au namespace définit pour notre groupe sur kubernetes, càd `l6grt`.

Pods

Le type de volume est maintenant `secret`.

Conclusion

Problèmes rencontré

NAME	READY	STATUS	RESTARTS	AGE
pod/api-deploy-86d8969586-b8v5c	0/1	ImagePullBackOff	0	87s
pod/api-deploy-86d8969586-sxb9x	0/1	ImagePullBackOff	0	87s
pod/frontend-deploy-785865d54b-w66sq	1/1	Running	0	100s
pod/frontend-deploy-785865d54b-x4grp	0/1	ImagePullBackOff	0	100s
pod/redis-deploy-6fbcc669d9-tq6kc	0/1	ImagePullBackOff	0	93s

Pour résoudre cette erreur, nous avons simplement effectuer la commande
`kubectl describe pod/api-deploy-86d8969586-b8v5c`

pour pouvoir voir les events qui correspondait. Nous avons donc pu constater que l'erreur suivante :

Failed to pull image "icclabcna/ccp2-k8s-todo-fron: toomanyrequests: You have reached your pull rate limit. You may increase the limit by authenticating and upgrading

Comme l'a décrit M.Poulard dans le teams, il fallait rajouter les lignes suivantes dans les fichiers de configuration YAML des deployment :

`imagePullPolicy: IfNotPresent`

Ainsi, la création de tous les pods se fait correctement.

NAME	READY	STATUS	RESTARTS	AGE
pod/api-deploy-6779c75f68-h64dj	1/1	Running	2	9m38s
pod/api-deploy-6779c75f68-smk99	1/1	Running	2	9m40s
pod/frontend-deploy-7dd4c876-gpmht	1/1	Running	0	9m49s
pod/frontend-deploy-7dd4c876-l7nt8	1/1	Running	0	9m47s
pod/redis-deploy-869b9b86b7-jx219	1/1	Running	0	9m33s

Nous n'avons pas rencontré d'autres problèmes particulier lors des taches de ce laboratoire.