

Secret

# Utilisation

- Objet permettant la protection des données sensibles
  - clé privée, mot de passe, chaîne de connexion à un service tiers, ...
- Évite de définir ces informations dans les images ou spécifications
- Donne un meilleur contrôle sur leur utilisation
- Créé par un utilisateur ou par le système (clés d'accès à l'API)
- Utilisé dans un Pod
  - via un volume monté dans un ou plusieurs containers
  - via kubelet lors de la récupération de l'image
- Stocké dans etcd (encryption expérimentale)

# Différents types

- generic
  - Secret créé à partir d'un fichier, d'un répertoire ou d'une valeur littérale
- docker-registry
  - Secret utilisé pour l'authentification à un Docker registry
- TLS
  - Secret utilisé pour la gestion des clés (PKI)

# Secret de type generic : création

```
# Création de fichiers contenant les credentials de connexion à un service tiers
$ echo -n "admin" > ./username.txt
$ echo -n "45fe3efa" > ./password.txt

# Création de l'objet Secret avec kubectl
$ kubectl create secret generic service-creds --from-file=./username.txt --from-file=./password.txt
secret "service-creds" created

# Création depuis des valeurs littérales
$ kubectl create secret generic service-creds2 \
  --from-literal=username=admin --from-literal=password=45fe3efa
secret "service-creds2" created

# Liste des Secrets présents
$ kubectl get secrets
```

NAME	TYPE	DATA	AGE
default-token-ps46p	kubernetes.io/service-account-token	3	82d
service-creds	Opaque	2	3s
service-creds2	Opaque	2	3m

Secret généré par kubernetes pour permettre aux services internes d'accéder à l'API server

# Secret de type generic : création

```
# Inspection du contenu de l'objet
$ kubectl describe secrets/service-creds
```

```
Name:          service-creds
Namespace:     default
Labels:        <none>
Annotations:   <none>
```

```
Type: Opaque
```

```
Data
```

```
====
```

```
password.txt: 8 bytes
username.txt: 5 bytes
```

```
# Détails de la spécification de l'objet
$ kubectl get secrets service-creds -o yaml
```

```
apiVersion: v1
data:
  password.txt: NDVmZTNlZmE=
  username.txt: YWRtaW4=
kind: Secret
metadata:
  creationTimestamp: 2018-02-28T22:37:47Z
  name: service-creds
  namespace: default
  resourceVersion: "589455"
  selfLink: /api/v1/namespaces/default/secrets/service-creds
  uid: ffd1b396-1cd7-11e8-ba7f-080027f0e385
type: Opaque
```



Encodé en base64

```
$ echo "NDVmZTNlZmE=" | base64 -D
45fe3efa
$ echo "YWRtaW4=" | base64 -D
admin
```

# Secret de type generic : création

```
# Conversion de la donnée sensible en base64
$ echo -n "mongodb://admin:45fe3efa@mgserv1.org/mgmt" | base64
bW9uZ29kYjovL2FkbWluOjQ1ZmUzZWZhQG1nc2VydjEub3JnL21nbXQ=

# Spécification de l'objet Secret
$ cat mongo-creds.yaml
apiVersion: v1
kind: Secret
metadata:
  name: mongo-creds
data:
  mongoURL: bW9uZ29kYjovL2FkbWluOjQ1ZmUzZWZhQG1nc2VydjEub3JnL21nbXQ=

# Création de l'objet Secret
$ kubectl create -f ./mongo-creds.yaml
secret "mongo-creds" created
```



# Secret de type generic : utilisation (volume 1)

```
$ cat pod-secret-volume-1.yaml
```

```
apiVersion: v1
kind: Pod
metadata:
  name: alpine
spec:
  containers:
  - name: alpine
    image: alpine
    command:
    - "sleep"
    - "10000"
    volumeMounts:
    - name: creds
      mountPath: "/etc/creds"
      readOnly: true
  volumes:
  - name: creds
    secret:
      secretName: mongo-creds
```

Définition d'un volume basé sur le Secret mongo-creds

# Secret de type generic : utilisation (volume 1)

```
$ cat pod-secret-volume-1.yaml
apiVersion: v1
kind: Pod
metadata:
  name: alpine
spec:
  containers:
  - name: alpine
    image: alpine
    command:
    - "sleep"
    - "10000"
    volumeMounts:
    - name: creds
      mountPath: "/etc/creds"
      readOnly: true
  volumes:
  - name: creds
    secret:
      secretName: mongo-creds
```



Montage du volume dans le container sur le point de montage spécifié

Définition d'un volume basé sur le Secret mongo-creds



# Secret de type generic : utilisation (volume 1)

```
# Création du Pod alpine
$ kubectl create -f pod-secret-volume-1.yaml
pod "alpine" created

# Lancement d'un shell interactif dans le container alpine
$ kubectl exec -ti alpine -- sh
/ # cat /etc/creds/mongoURL
mongodb://admin:45fe3efa@mgserve1.org/mgmt
```

La clé **mongoURL** spécifiée dans le Secret est disponible dans le point de montage **/etc/creds**

# Secret de type generic : utilisation (volume 2)

```
$ cat pod-secret-volume-2.yaml
apiVersion: v1
kind: Pod
metadata:
  name: alpine
spec:
  containers:
  - name: alpine
    image: alpine
    command:
    - "sleep"
    - "10000"
    volumeMounts:
    - name: creds
      mountPath: "/etc/creds"
      readOnly: true
  volumes:
  - name: creds
    secret:
      secretName: service-creds
      items:
      - key: username.txt
        path: service/auth
      - key: password.txt
        path: service/pass
```

Définition d'un volume basé sur le Secret service-creds  
On précise les path relatifs où les données seront montées

# Secret de type generic : utilisation (volume 2)

```
$ cat pod-secret-volume-2.yaml
apiVersion: v1
kind: Pod
metadata:
  name: alpine
spec:
  containers:
  - name: alpine
    image: alpine
    command:
    - "sleep"
    - "10000"
    volumeMounts:
    - name: creds
      mountPath: "/etc/creds"
      readOnly: true
  volumes:
  - name: creds
    secret:
      secretName: service-creds
      items:
      - key: username.txt
        path: service/user
      - key: password.txt
        path: service/pass
```



Montage du volume dans le container sur le point de montage spécifié

Définition d'un volume basé sur le Secret service-creds  
On précise les path relatifs ou les clés seront montées

# Secret de type generic : utilisation (volume 2)

```
# Création du Pod alpine
$ kubectl create -f pod-secret-volume-2.yaml
pod "alpine" created

# Lancement d'un shell interactif dans le container api
$ kubectl exec -ti alpine -- sh
/ # cat /etc/creds/service/user
admin
/ # cat /etc/creds/service/pass
45fe3efa
```



Les valeurs des clés **username.txt** et **password.txt** sont montées dans les paths spécifiés sous le point de montage **/etc/creds**

# Secret de type generic : utilisation (env)

```
$ cat pod-secret-env.yaml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: alpine
```

```
spec:
```

```
  containers:
```

```
  - name: alpine
```

```
    image: alpine
```

```
    command:
```

```
    - "sleep"
```

```
    - "10000"
```

```
  env:
```

```
  - name: MONGO_URL
```

```
    valueFrom:
```

```
      secretKeyRef:
```

```
        name: mongo-creds
```

```
        key: mongoURL
```

Définition de la variable d'environnement MONGO\_URL

# Secret de type generic : utilisation (env)

```
$ cat pod-secret-env.yaml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: alpine
```

```
spec:
```

```
  containers:
```

```
  - name: alpine
```

```
    image: alpine
```

```
    command:
```

```
    - "sleep"
```

```
    - "10000"
```

```
  env:
```

```
  - name: MONGO_URL
```

```
    valueFrom:
```

```
      secretKeyRef:
```

```
        name: mongo-creds
```

```
        key: mongoURL
```

Définition de la variable d'environnement MONGO\_URL

Lecture de la valeur depuis la key mongoURL définie dans le Secret mongo-creds



# Secret de type generic : utilisation (env)

```
# Création du Pod api
$ kubectl create -f pod-secret-env.yaml
pod "api" created

# Lancement d'un shell interactif dans le container api
$ kubectl exec -ti alpine -- sh
/ # env | grep MONGO
MONGO_URL=mongodb://admin:45fe3efa@mgserve1.org/mgmt
```

La clé **mongoURL** spécifiée dans le Secret est disponible dans la variable d'environnement du container alpine

# Secret de type docker-registry : création

- Permet de s'identifier sur un registry Docker
- Récupération des images privées

```
$ kubectl create secret docker-registry registry-creds \  
  --docker-server=REGISTRY_FQDN --docker-username=USERNAME --docker-password=PASSWORD --docker-email=EMAIL  
  
$ kubectl get secret registry-creds -o yaml  
apiVersion: v1  
data:  
  .dockercfg: eyJhdXRocyI6eyJodHR...QVWtRPSJ9fX0=  ← Crédentials encodés en base64  
kind: Secret  
metadata:  
  creationTimestamp: 2018-03-02T22:26:14Z  
  name: registry-creds  
  namespace: default  
  resourceVersion: "626790"  
  selfLink: /api/v1/namespaces/default/secrets/registry-creds  
  uid: b7b70613-1e68-11e8-ba7f-080027f0e385  
type: kubernetes.io/dockercfg
```

# Secret de type docker-registry : utilisation

```
# Pod utilisant une image privée
```

```
$ cat pod-private-image.yaml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: private-image
```

```
spec:
```

```
  containers:
```

```
  - name: api
```

```
    image: my_private_image
```

```
  imagePullSecrets:
```

```
  - name: registry-creds
```

← Le Secret nommé *registry-creds* est utilisé pour le téléchargement d'une image du repository privé

```
$ kubectl create -f pod-private-reg.yaml
```

```
pod "private-reg" created
```

# Secret de type TLS : création

- Gestion des PKI
- Créé à partir d'un couple clé publique / clé privée

```
# Création du couple de clés
$ openssl req -newkey rsa:2048 -nodes -keyout key.pem -x509 -days 365 -out cert.pem

# Création du Secret à partir des clés
$ kubectl create secret tls domain-pki --cert cert.pem --key key.pem

$ kubectl get secret domain-pki -o yaml
apiVersion: v1
data:
  tls.crt: LS0tLS1CRUdJT...GSUNBVEUtLS0tLQo=
  tls.key: LS0tLS1CRUdJT...TESBLRVktLS0tLQo=
kind: Secret
Metadata:
  name: domain-pki
  ...
type: kubernetes.io/tls
```

← Clés encodées en base64

# Secret de type TLS : utilisation

```
$ cat pod-secret-tls.yaml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: proxy
```

```
spec:
```

```
  containers:
```

```
  - name: proxy
```

```
    image: nginx:1.12.2
```

```
    volumeMounts:
```

```
    - name: tls
```

```
      mountPath: "/etc/ssl/certs/"
```

```
  volumes:
```

```
  - name: tls
```

```
    secret:
```

```
      secretName: domain-pki
```

Définition d'un volume basé sur le Secret domain-pki



# Secret de type TLS : utilisation

```
$ cat pod-secret-tls.yaml
```

```
apiVersion: v1
```

```
kind: Pod
```

```
metadata:
```

```
  name: proxy
```

```
spec:
```

```
  containers:
```

```
  - name: proxy
```

```
    image: nginx:1.12.2
```

```
    volumeMounts:
```

```
    - name: tls
```


```
      mountPath: "/etc/ssl/certs/"
```

```
  volumes:
```

```
  - name: tls
```

```
    secret:
```

```
      secretName: domain-pki
```



Montage du volume dans le container sur le point de montage spécifié

Définition d'un volume basé sur le Secret domain-pki



# Secret de type TLS : utilisation

```
# Lancement du Pod
$ kubectl create -f pod-secret-tls.yaml
pod "proxy" created

# Lancement d'un shell interactif dans le container
$ kubectl exec -ti proxy -- sh
/ # ls /etc/ssl/certs
tls.crt  tls.key
```