Projet AppScore - Sécurité

Utiliser un ServiceAccount dédié pour au moins un des microservices, avec des droits restreints (RBAC).

```
kubectl get pod -l app=webapp -o
jsonpath='{.items[0].spec.serviceAccountName}'
```

```
PS C:\Ynov\M2\Conteneurisation\appscore4> kubectl get pod -l app=webapp -o jsonpath='{.items[0].spec.serviceAccountName}'
sa-webapp
```

"Ici, on voit que mon Pod `webapp` n'utilise plus le `default` ServiceAccount. Il a sa propre identité nommée `sa-webapp`."

Prouver que les droits sont restreints :

```
PS C:\Ynov\M2\Conteneurisation\appscore4> kubectl get rolebinding bind-webapp-role
NAME               ROLE                        AGE
bind-webapp-role   Role/role-webapp-lecteur   8s
```

```
PS C:\Ynov\M2\Conteneurisation\appscore4> kubectl auth can-i list pods --as=system:serviceaccount:default:sa-webapp
yes
```

Ce que ce "yes" signifie concrètement :
Cela veut dire que la chaîne de confiance Kubernetes est complète :

Le ServiceAccount (sa-webapp) est reconnu.
Le Role est valide.
Le RoleBinding fait bien le pont entre les deux.

Désormais, n'importe quel Pod qui utilise serviceAccountName: sa-webapp aura le droit de lister les pods, mais rien d'autre.

Mettre en place au moins une NetworkPolicy pour limiter les communications :
Par exemple : seule web peut appeler identity.api, seules les API peuvent parler à sql.data, etc.

```
PS C:\Ynov\M2\Conteneurisation\appscore4> kubectl describe netpol allow-api-to-sql
Name:         allow-api-to-sql
Namespace:    default
Created on:   2026-01-26 22:45:12 +0100 CET
Labels:       <none>
Annotations:  <none>
Spec:
  PodSelector:     app=sql-data
  Allowing ingress traffic:
    To Port: 5432/TCP
    From:
      PodSelector: app=jobs-api
    From:
      PodSelector: app=applicants-api
  Not affecting egress traffic
  Policy Types: Ingress
```

La règle qu'on vient de mettre en place s'appelle le "Principe de l'Isolation Réseau".

Pour faire simple : c'est comme si tu avais installé une porte blindée avec un lecteur de badge devant ta base de données (sql-data).

Voici exactement ce que fait cette règle :

1. Elle ferme toutes les portes par défaut
   Dès que tu appliques une NetworkPolicy sur un Pod (ici ton Pod sql-data), Kubernetes passe en mode "Tout ce qui n'est pas autorisé est interdit".

Avant la règle : N'importe qui pouvait parler à ta base.

Après la règle : Plus personne ne peut lui parler, sauf ceux qui ont le bon badge.

2. Elle définit la "Liste Blanche" (Whitelist)
   Dans le fichier, on a écrit une liste d'invités autorisés à entrer. Seuls les Pods qui portent ces étiquettes (labels) peuvent passer :

Les Pods avec le label app: jobs-api

Les Pods avec le label app: applicants-api

3. Elle limite le tunnel (Le Port)
   On ne se contente pas de dire "qui" peut entrer, on dit aussi "par où". On a ouvert uniquement le port 5432 (le port standard SQL). Même un Pod autorisé ne pourra pas essayer de se connecter sur un autre port.

Schéma de ce qu'il se passe maintenant :
Scénario A : Ton application webapp essaie de contacter sql-data.

Résultat : BLOQUÉ. Elle n'est pas dans la liste des invités.

Scénario B : Un pirate réussit à entrer dans ta webapp. Il essaie de scanner ta base de données.

Résultat : BLOQUÉ. Il ne voit même pas que la base existe.

Scénario C : Ton service jobs-api fait une requête à la base.

Résultat : AUTORISÉ. Le badge correspond.

```
PS C:\Ynov\M2\Conteneurisation\appscore4> kubectl describe netpol allow-api-to-sql
Name:         allow-api-to-sql
Namespace:    default
Created on:   2026-01-26 22:45:12 +0100 CET
Labels:       <none>
Annotations:  <none>
Spec:
  PodSelector:     app=sql-data
  Allowing ingress traffic:
    To Port: 5432/TCP
    From:
      PodSelector: app=service-api-jobs
    From:
      PodSelector: app=service-api-identity
    From:
      PodSelector: app=applicants-api
  Not affecting egress traffic
  Policy Types: Ingress
```