Lab 5b - Secure network traffic inside your cluster

Introduction

In this Lab you configure the YADA application to ensure API is only accessible from the Web frontend. Other pods hosted in your cluster (from other projects, teams etc.) must not access your API.

To do that you will create a `NetworkPolicy` with the corresponding rules. You will also test the rule is functioning and the application is still working as it should.

- Expected Lab duration: 30 minutes

Challenge yourself (OPTIONAL)

If you want to challenge yourself, try to perform the following actions on your own:

1. Execute a curl command from the `Web` pod to your `API service` health check URL: `http://api.api:8080/api/healthcheck`
2. Add a network policy on the `API` pods to only allow traffic coming from (AND rule):
    1. Pods in a namespace with a `app: yada` label
    2. Pods having the `run: web` label
3. Make adjustments in your environment to make the first test working again (based on your previous rule)
4. Run another pod that doesn't comply with the rule and verify traffic doesn't reach your API

Exercise 1: Review and Deploy a Network Policy

To use Network Policy in AKS, your network plugin must be enabled for that. When we created the cluster, this feature has been enabled. In case it was not enabled, you need to recreate the cluster as it's not possible to change this setting after cluster creation.

1. Open **Azure Cloud Shell**

2. Execute the following command to verify your Web pods can successfully connect to your API (`-m 2` parameter is used to exit after 2 seconds):

    ```
    kubectl exec -n web deployments/web -- curl
    http://api.api:8080/api/healthcheck -s -m 2
    ```

3. Download the YAML files of the lab in your Cloud Shell, open the files to understand them or execute them directly:

    ```
    wget https://raw.githubusercontent.com/jbpaux/aks-infra-
    training/main/yaml/lab5b/api-networkpolicy.yml
    kubectl apply -f api-networkpolicy.yml -n api
    ```

4. Re-execute the same check as in task 2. It should now fail. Analyze the `NetworkPolicy` object to understand what differ in your environment. Any guess?

5. I hope you noticed the rule only allow traffic coming from pods in having `run: web` label in a namespace having `app: yada` label. Your namespace doesn't currently have the label. Let's add it

```
kubectl describe networkpolicy -n api api-networkpolicy
kubectl get pods -n web --show-labels
kubectl describe ns web
kubectl label ns web app=yada
kubectl describe ns web
```

6. Execute for the last time the curl command to ensure your application can now communicate successfully again.

7. This command will create a pod in the `default` namespace named `curl` that will execute the same command and then get removed. As we don't have the correct labels, it should fail:

```
kubectl run -n default --image curlimages/curl --rm --restart Never -i --
curl http://api.api:8080/api/healthcheck -s -m 5
```