# Practice M8: Exam Preparation (Challenge)

## Exam Plot

Should you want to reproduce the environment, you will have to prepare the following set of machines:

* **Exam station** is with **kubectl** + **helm** + **kustomize** installed
* **Mars cluster** – two machines – working cluster with **Flannel** as a pod network plugin and **HAProxy** as ingress controller
* **Jupiter cluster** – three machines – working cluster with **WeaveNet** as a pod network plugin
* **Venus cluster** – two machines – **broken** cluster with **Antrea** as a pod network plugin (should be installed as part of the solution)

All clusters are **Kubernetes 1.28.3** and all their nodes are with **2 vCPU / 2 GB RAM / 32 GB Disk / Debian 12**. The exam station is with **1 vCPU / 2 GB RAM**

There is the **exam** user which has **sudo** privileges on all machines and can authenticate to each one of them via SSH

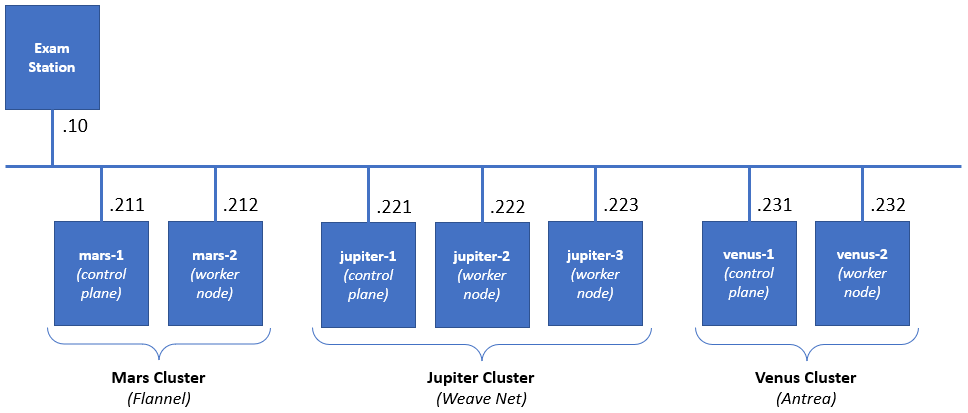
If you are constrained by the available resources, you can spin up the clusters one by one, as the tasks are grouped by cluster. You can even omit the exam station and use your host to control and work with the clusters

Check the published archive for the setup files (folder **1-setup**). There is a separate folder (**2-files**) that contains the source or input files referenced in the tasks. Its content is expected to be copied in **/files** folder of the station

## Sample Exam

### Infrastructure

You will have to accomplish a set of tasks in the following infrastructure



### Rules

Be sure to **follow strictly** the **naming** **conventions** specified in the checklist

Tasks execution order should not be derived from the order in which they are listed below. Please note that there are tasks that depend on the successful completion of one or more other tasks

Usually, all steps could be achieved by following different paths and using different tools. In the end, not the means, but the **results** are being **measured**, **except stated otherwise**

### Tasks checklist

#### Mars Cluster [18 pts]

* (T101 / 3 pts) There is the **animals** namespace. There are two pairs of pod and service. You are expected to create an ingress resource (using the available ingress controller and class) that
  + Will be in the same namespace and named **pets-ingress**
  + Will serve the **pets.lab** host
  + Path **/cat** to be redirected to **cat-svc** service and **/dog** to the **dog-svc**
  + Store it in **/files/mars/t101-out.yaml** and be sure to deploy it
* (T102 / 3 pts) Explore the **tiger** namespace. There is a pod that is not in running state, but it should be. Its manifest is **/files/mars/t102-in.yaml**. Your mission, should you accept it, is to
  + Correct the issue(s) and reflect this in a new manifest **/files/mars/t102-out.yaml**
  + Make sure that the pod is in running state and doesn’t restart periodically because of a probe
* (T103 / 4 pts) Templating is good and necessary technique. We have a simple manifest **(/files/mars/t103-in.yaml**) which we want to be able to easily deploy in production (**blue**) and in test (**green**). Using the **kustomize** application, you must prepare a set of folders and files in the **/files/mars/t103** folder that allows
  + Base (without any changes) deployment and deployment to both environments
  + The **blue** deployment should increase the **replicas** to **3**, use the **blue** tag, and runs on port **31103**
  + The **green** deployment should use the **green** image tag and runs on port **32103**
  + Make sure that blue and green are deployed (but not the base)
* (T104 / 2 pts) Explore the **cherry** namespace. There is a deployment that is failing. Your mission is to
  + Find the reason for this and write it down *(****type of the object:name of the object****, for example* ***limit:banana****)* in the **/files/mars/t104-reason.txt** file
  + Correct the situation by changing the offending parameter of the deployment to comply
* (T105 / 2 pts) Explore the pod manifest in **/files/mars/t105-in.yaml** file and
  + Create a new one (**/files/mars/t105-out.yaml**) that wraps the pod template in a **CronJob** named **five-job**
  + Set it to run **every 5 minutes** and deploy it
* (T106 / 4 pts) There is the **fortress** namespace. It is empty. Your mission is to
  + Create a **ServiceAccount** named **observer**
  + Create a **Role** named **looknotouch** that allows only **get** on **pods**
  + Create a **RoleBinding** named **looknotouch** that binds the role to the service account
  + Modify the pod manifest **/files/mars/t106-in.yaml** to run the pod with the observer service account, store the new version in **/files/mars/t106-out.yaml** and deploy it

#### Jupiter Cluster [25 pts]

* (T201 / 4 pts) There are three namespaces – **apple**, **orange**, and **apricot**. In all three, there is a pair of a pod and service. There aren’t any restrictions. You should correct this:
  + Add a network policy that will limit the **ingress access** to the pods in the **apple** namespace to connections, coming only from pods in the **orange** namespace
* (T202 / 5 pts) We all know that using **Helm** charts is both fun and easy. So, let’s spin up one chart
  + Use the **artifacthub.io** and find the **Apache HTTP** chart provided by **Bitnami** and **add the repository**
  + Then **install** the chart as **exam-httpd** release in the **kiwi** namespace (create it if not existing)
  + Make sure that using the chart’s parameters it is set to use a service of type **NodePort** and to listen for HTTP request on port **32202**
  + Create a **ConfigMap** named **exam-httpd-cm** in the same namespace that contains an **index.html** file with the following text **Helm+Kubernetes=Fun** and attach it to the release
* (T203 / 4 pts) There is a pod in the **cucumber** namespace that is consuming a secret in the same namespace. You are expected to:
  + Find the unencoded value of the secret and save it as **/files/jupiter/t203-secret.txt**
  + Then change the secret to **Cucumbers are green**
* (T204 / 2 pts) Add a new label (**exam**) to both worker nodes (**jupiter-2** and **jupiter-3**):
  + Set it to **slow** for **jupiter-2**
  + Set it to **fast** for **jupiter-3**
* (T205 / 1 pts) Explore the manifest **/files/jupiter/t205-in.yaml**
  + Modify it in such a way that if deployed, the workload to go on the node with label **exam** set to **fast** and save the new manifest as **/files/jupiter/t205-out.yaml**
  + Deploy it to the cluster
* (T206 / 1 pts) Explore the manifest **/files/jupiter/t206-in.yaml**
  + Modify it in such a way that if deployed, the workload to go on the node named **jupiter-2** and save the new manifest as **/files/jupiter/t206-out.yaml**
  + Deploy it to the cluster
* (T207 / 4 pts) Explore the **banana** namespace. There should be a pair of a pod and service. They are created out of the **/files/jupiter/t207-in.yaml** manifest. The problem is that the pod (**banana-pod-1**) is not in running state and the service (**banana-svc**) does not have any endpoints. Your mission is to:
  + Correct these issues and save the changes as **/files/jupiter/t207-out.yaml** manifest
  + Make sure that the deployed objects are in a good shape (they reflect the corrections)
  + Add a second pod, in the new manifest file, of the same type but change the image tag to green and name it **banana-pod-2**
  + Make sure that the new pod is present in the service endpoints list
* (T208 / 4 pts) Explore the manifest **/files/jupiter/t208-in.yaml** and
  + Extend it to also include
    - A definition of a namespace named **cherry**
    - and a definition of a service named **cherry-svc** of type **NodePort** and port set to **32208**
    - and save it as **/files/jupiter/t208-out.yaml**
  + Deploy the manifest

#### Venus Cluster [17 pts]

* (T301 / 5 pts) Install the missing system components (**kubeadm**, **kubelet**, and **kubectl**) on the **venus-2** node and make sure that their version is aligned with the version installed on the **venus-1** node
* (T302 / 3 pts) Join the **venus-2** node to the **Venus Cluster**
* (T303 / 2 pts) Deploy **Antrea** pod network plugin on the **Venus** **Cluster**
* (T304 / 3 pts) Modify the configuration of the **Venus** **Cluster** in such a way to allow workload to be placed on the **control plane** node
* (T305 / 4 pts) Explore the manifest **/files/venus/t305-in.yaml** and
  + Change it in such a way (save it under **/files/venus/t305-out.yaml**) that the described pod is **deployed on every node of the cluster**
  + Deploy the resulting manifest