

Lab 5 K8s

1. Where is the default kubeconfig file located in the current environment

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~$ cat ~/.kube/config
apiVersion: v1
clusters:
- cluster:
    certificate-authority: /home/shehab-gamal/.minikube/ca.crt
    extensions:
    - extension:
        last-update: Sat, 15 Feb 2025 14:27:36 EET
        provider: minikube.sigs.k8s.io
        version: v1.35.0
    name: cluster info
```

or you can use \$KUBECONFIG environment variable if it's set.

2. How many clusters are defined in the default kubeconfig

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl config get-clusters
NAME
minikube
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$
```

3. What is the user configured in the current context

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl config get-contexts
CURRENT  NAME      CLUSTER  AUTHINFO  NAMESPACE
*        minikube  minikube minikube  default
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$
```

4. Create a Persistent Volume with the given specification.

- Volume Name: pv-log
- Storage: 100Mi
- Access Modes: ReadWriteMany
- Host Path: /pv/log

```
io.k8s.api.core.v1.PersistentVolume (v1@persistentvolume.json)
1  apiVersion: v1
2  kind: PersistentVolume
3  metadata:
4    | name: pv-log
5  spec:
6    | capacity:
7    |   storage: 100Mi
8    | accessModes:
9    |   - ReadWriteMany
10   | persistentVolumeReclaimPolicy: Retain
11   | hostPath:
12   |   path: "/pv/log"
```

5. Create a Persistent Volume Claim with the given specification.

→ Volume Name: claim-log-1

→ Storage Request: 50Mi

→ Access Modes: ReadWriteMany

```
lab5 > ! pvc-log.yml > {} spec > {} resources >
io.k8s.api.core.v1.PersistentVolumeClaim (v1)
1  apiVersion: v1
2  kind: PersistentVolumeClaim
3  metadata:
4    name: claim-log-1
5  spec:
6    accessModes:
7      - ReadWriteMany
8    resources:
9      requests:
10     storage: 50Mi
```

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl get pvc
NAME          STATUS    VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS   VOLUMEATTRIBUTESCLASS   AGE
claim-log-1   Bound     pvc-c75cb132-7286-4ab6-b526-df61c6fe2c05   50Mi       RWX            standard       <unset>            28m
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$
```

6. Create a webapp pod to use the persistent volume claim as its storage.

→ Name: webapp

→ Image Name: nginx

→ Volume: PersistentVolumeClaim=claim-log-1

→ Volume Mount: /var/log/nginx

```
lab5 > ! webapp-pod.yml > {} spec > [ ] volumes > {} 0 > {} p
io.k8s.api.core.v1.Pod (v1@pod.json)
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: webapp
5  spec:
6    containers:
7      - name: nginx
8        image: nginx
9        volumeMounts:
10       - mountPath: "/var/log/nginx"
11         name: log-volume
12    volumes:
13      - name: log-volume
14        persistentVolumeClaim:
15          claimName: claim-log-1
```

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl describe pod webapp | grep ClaimName
ClaimName: claim-log-1
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$
```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/Lab5$ kubectl exec -it webapp -- /bin/sh
# ls -l var/log/nginx
total 4
-rw-r--r-- 1 root root 0 Feb 15 12:57 access.log
-rw-r--r-- 1 root root 488 Feb 15 12:57 error.log
#

```

7. Create a pod named volume-share-datacenter.

For first container, use image centos:latest, container should be named as volume-container-datacenter-1, and run a command '/bin/bash', '-c' and 'sleep 10000'. Volume volume-share should be mounted at path /tmp/news.

For second container, use image centos:latest, container should be named as volume-container-datacenter-2, and run a command '/bin/bash', '-c' and 'sleep 10000'. Volume volume-share should be mounted at path /tmp/cluster.

Volumes to be named as volume-share and use emptyDir: { }.

After creating the pod, exec into the first container volume-container-datacenter-1, and create a file news.txt with content Welcome from datacenter-1! under the mount path of first container /tmp/news.

The file news.txt should be present under the mounted path /tmp/cluster of second container volume-container-datacenter-2 as they are using shared Volumes.

```

lab5 > ! volume-share-datacenter.yml > {} spec > [ ] containers > {} 1 > [ ] command
io.k8s.api.core.v1.Pod (v1@pod.json)
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: volume-share-datacenter
5  spec:
6    containers:
7      - name: volume-container-datacenter-1
8        image: centos:latest
9        command: ["/bin/bash", "-c", "sleep 10000"]
10       volumeMounts:
11         - mountPath: "/tmp/news"
12           name: volume-share
13       - name: volume-container-datacenter-2
14         image: centos:latest
15         command: ["/bin/bash", "-c", "sleep 10000"]
16         volumeMounts:
17           - mountPath: "/tmp/cluster"
18             name: volume-share
19     volumes:
20       - name: volume-share
21         emptyDir: {}
22

```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/Lab5$ kubectl exec -it volume-share-datacenter -c volume-container-datacenter-1 -- /bin/bash
[root@volume-share-datacenter /]# echo "Welcome from datacenter-1!" > /tmp/news/news.txt
[root@volume-share-datacenter /]# exit
exit
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/Lab5$

```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/Lab5$ kubectl exec -it volume-share-datacenter -c volume-container-datacenter-2 -- /bin/bash
[root@volume-share-datacenter /]# cat /tmp/news/news.txt
Welcome from datacenter-1!
[root@volume-share-datacenter /]#

```

8. Create a pod named webserver.

Create an emptyDir volume name: shared-logs.

Create two containers from nginx and ubuntu images with latest tag only and remember to mention tag i.e nginx:latest, nginx container name should be Nginx-container and ubuntu container name should be sidecar-container on webserver pod. Add command on sidecar-container "sh", "-c", "while true; do cat /var/log/nginx/access.log /var/log/nginx/error.log; sleep 30; done"

Mount volume /var/log/nginx on both containers, all containers should be up and Running.

```
io.k8s.api.core.v1.Pod (v1@pod.json)
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: webserver
5  spec:
6    containers:
7      - name: nginx-container
8        image: nginx:latest
9        volumeMounts:
10         - mountPath: "/var/log/nginx"
11           name: shared-logs
12      - name: sidecar-container
13        image: ubuntu:latest
14        command: ["sh", "-c", "while true; do cat /var/log/nginx/access.log /var/log/nginx/error.log; sleep 30; done"]
15        volumeMounts:
16         - mountPath: "/var/log/nginx"
17           name: shared-logs
18    volumes:
19      - name: shared-logs
20        emptyDir: {}
```

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl get pods | grep webserver
webserver          2/2      Running    0          77s
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$
```

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl logs webserver -c sidecar-container
2025/02/15 13:43:48 [notice] 1#1: using the "epoll" event method
2025/02/15 13:43:48 [notice] 1#1: nginx/1.27.4
2025/02/15 13:43:48 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/02/15 13:43:48 [notice] 1#1: OS: Linux 5.10.207
2025/02/15 13:43:48 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025/02/15 13:43:48 [notice] 1#1: start worker processes
2025/02/15 13:43:48 [notice] 1#1: start worker process 29
2025/02/15 13:43:48 [notice] 1#1: start worker process 30
2025/02/15 13:43:48 [notice] 1#1: using the "epoll" event method
2025/02/15 13:43:48 [notice] 1#1: nginx/1.27.4
2025/02/15 13:43:48 [notice] 1#1: built by gcc 12.2.0 (Debian 12.2.0-14)
2025/02/15 13:43:48 [notice] 1#1: OS: Linux 5.10.207
```

9. Create a new service account with the name pvviewer. Grant this Service account access to list all PersistentVolumes in the cluster by creating an appropriate cluster role called pvviewer-role and ClusterRoleBinding called pvviewer-role-binding.

```
lab5 > ! pvviewer.yml > ...
io.k8s.api.core.v1.ServiceAccount (v1@serviceaccount.json)
1  apiVersion: v1
2  kind: ServiceAccount
3  metadata:
4    name: pvviewer
```

```
lab5 > ! pvviewer-role.yml > [ ] rules > {} 0
io.k8s.api.rbac.v1.ClusterRole (v1@clusterrole.json)
1  apiVersion: rbac.authorization.k8s.io/v1
2  kind: ClusterRole
3  metadata:
4    name: pvviewer-role
5  rules:
6    - apiGroups: ["" ]
7      resources: ["persistentvolumes"]
8      verbs: ["get", "list"]
```



```
lab5 > ! pvviewer-pod.yml > {} spec > [ ] containers >
io.k8s.api.core.v1.Pod (v1@pod.json)
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: my-pd
5  spec:
6    serviceAccountName: pvviewer
7    containers:
8      - image: nginx
9        name: nginx
```

```
lab5 > ! pvviewer-role-bind.yml > [ ] subjects > {} 0 > apiGroup
io.k8s.api.rbac.v1.ClusterRoleBinding (v1@clusterrolebinding.json)
1  apiVersion: rbac.authorization.k8s.io/v1
2  kind: ClusterRoleBinding
3  metadata:
4    name: pvviewer-role-bind
5  roleRef:
6    kind: ClusterRole
7    name: pvviewer-role
8    apiGroup: rbac.authorization.k8s.io
9  subjects:
10 - kind: ServiceAccount
11   name: pvviewer
12   namespace: default
13   apiGroup: ""
```

```
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl exec -it my-pd -- sh
# kubectl get pv
NAME          CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS    CLAIM                STORAGECLASS
pv-log        100Mi     RWX           Retain          Available
pvc-c75cb132-7286-4ab6-b526-df61c6fe2c05  50Mi      RWX           Delete          Bound          default/claim-log-1  standard
```

10. Create a ConfigMap named nginx-config with the following Nginx configuration: nginx.conf:

Create a Pod named nginx-pod that:

- Uses the Nginx image, & mounts the ConfigMap as a volume.
- Ensures that the nginx.conf file from the ConfigMap is available at /etc/nginx/nginx.conf.

Verify the Pod by:

- Checking the pod logs.
- Executing a command inside the pod to test the custom configuration.

```
io.k8s.api.core.v1.Pod (v1@pod.json)
1  apiVersion: v1
2  kind: Pod
3  metadata:
4    name: nginx-pod
5  spec:
6    containers:
7      - name: nginx
8        image: nginx:latest
9        volumeMounts:
10       - name: config-volume
11         mountPath: /etc/nginx/
12    volumes:
13      - name: config-volume
14        configMap:
15          name: nginx-config
16
```

```
io.k8s.api.core.v1.ConfigMap (v1@configmap.json)
1  apiVersion: v1
2  kind: ConfigMap
3  metadata:
4    name: nginx-config
5  data:
6    nginx.conf: |
7      events {}
8
9      http {
10         server {
11             listen 80;
12             location / {
13                 return 200 'Custom Nginx Config Loaded!';
14             }
15         }
16     }
17
```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl exec -it nginx-pod -- sh
# cat /etc/nginx/nginx.conf
events {}

http {
    server {
        listen 80;
        location / {
            return 200 'Custom Nginx Config Loaded!';
        }
    }
}
#

```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl logs nginx-pod
/docker-entrypoint.sh: /docker-entrypoint.d/ is not empty, will attempt to perform configuration
/docker-entrypoint.sh: Looking for shell scripts in /docker-entrypoint.d/
/docker-entrypoint.sh: Launching /docker-entrypoint.d/10-listen-on-ipv6-by-default.sh
10-listen-on-ipv6-by-default.sh: info: /etc/nginx/conf.d/default.conf is not a file or does not exist
/docker-entrypoint.sh: Sourcing /docker-entrypoint.d/15-local-resolvers.envsh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/20-envsubst-on-templates.sh
/docker-entrypoint.sh: Launching /docker-entrypoint.d/30-tune-worker-processes.sh
/docker-entrypoint.sh: Configuration complete; ready for start up
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$

```

Deploy HaProxy

1. Create a namespace haproxy-controller-devops.

```

io.k8s.api.core.v1.Namespace (v1@namespace.json)
1  apiVersion: v1
2  kind: Namespace
3  metadata:
4    name: haproxy-controller-devops

```

```

shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$ kubectl get ns
NAME                                STATUS  AGE
default                            Active  2d2h
haproxy-controller-devops          Active  89s
iti                                 Active  84m
kube-node-lease                    Active  2d2h
kube-public                        Active  2d2h
kube-system                        Active  2d2h
shehab-gamal@shehab-gamal-Lenovo-ideapad-520-15IKB:~/Kubernetes-labs/lab5$

```

2. Create a ServiceAccount haproxy-service-account-devops under the same namespace.

```

io.k8s.api.core.v1.ServiceAccount (v1@serviceaccount.json)
1  apiVersion: v1
2  kind: ServiceAccount
3  metadata:
4    name: haproxy-service-account-devops
5    namespace: haproxy-controller-devops
6

```

3. Create a ClusterRole which should be named as haproxy-cluster-role-devops, to grant permissions "get", "list", "watch", "create", "patch", "update" to "Configmaps", "secrets", "endpoints", "nodes", "pods", "services", "namespaces", "events", "serviceaccounts".

```
1  apiVersion: rbac.authorization.k8s.io/v1
2  kind: ClusterRole
3  metadata:
4    name: haproxy-cluster-role-devops
5  rules:
6    - apiGroups: [ "", "networking.k8s.io", "apiextensions.k8s.io", "discovery.k8s.io" ]
7      resources:
8        - configmaps
9        - secrets
10       - endpoints
11       - nodes
12       - pods
13       - services
14       - namespaces
15       - events
16       - serviceaccounts
17       - ingresses
18       - ingressclasses
19       - endpointslices
20       - customresourcedefinitions
21     verbs:
22       - get
23       - list
24       - watch
25       - create
26       - patch
27       - update
```

4. Create a ClusterRoleBinding which should be named as haproxy-cluster-role-binding-devops under the same namespace. Define roleRef apiGro should be rbac.authorization.k8s.io, kind should be ClusterRole, name should be haproxy-cluster-role-devops and subjects kind should be ServiceAccount, name should be haproxy-service-account-devops and namespace should be haproxy-controller-devops.

```
lab5 > ! haproxy-clusterrole-bind.yml > {} roleRef
io.k8s.api.rbac.v1.ClusterRoleBinding (v1@clusterrolebinding.json)
1  apiVersion: rbac.authorization.k8s.io/v1
2  kind: ClusterRoleBinding
3  metadata:
4    name: haproxy-cluster-role-binding-devops
5  subjects:
6    - kind: ServiceAccount
7      name: haproxy-service-account-devops
8      namespace: haproxy-controller-devops
9  roleRef:
10   kind: ClusterRole
11   name: haproxy-cluster-role-devops
12   apiGroup: rbac.authorization.k8s.io
```


5. Create a backend deployment which should be named as backend-deployment-devops under the same namespace,
→ labels “run=ingress-default-backend”,
→ replicas=1,
→ container name=backend-container-devops,
→ image gcr.io/google_containers/defaultbackend:1.0 → containerPort=8080.

```
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: backend-deployment-devops
5    namespace: haproxy-controller-devops
6    labels:
7      run: ingress-default-backend
8  spec:
9    replicas: 1
10   selector:
11     matchLabels:
12       run: ingress-default-backend
13   template:
14     metadata:
15       labels:
16         run: ingress-default-backend
17     spec:
18       containers:
19         - name: backend-container
20           image: gcr.io/google_containers/defaultbackend:1.0
21           ports:
22             - containerPort: 8080
```

6. Create a service for backend which should be named as “service-backend-devops” under the same namespace, port should be named as port-backend.

```
io.k8s.api.core.v1.Service (v1@service.json)
1  apiVersion: v1
2  kind: Service
3  metadata:
4    name: service-backend-devops
5    namespace: haproxy-controller-devops
6  spec:
7    selector:
8      run: ingress-default-backend
9    ports:
10     - name: port-backend
11       port: 8080
12       targetPort: 8080
13
```

7. Create a deployment for frontend which should be named haproxy-ingress-devops under the same namespace. replicas=1, labels "run=haproxy-ingress". container name=ingress-container-devops service account=haproxy-service-account-devops image=haproxytech/kubernetes-ingress, give args as --default-backend-service=haproxy-controller-devops/service-backend-devops,

Resources

requests

→ cpu=500m

→ memory=50Mi

livenessProbe

→ httpGet path should be /health its port should be 1024.

The first port name should be http and its containerPort should be 80 the second port name should be https and its containerPort should be 443 third port name should be stat its containerPort should be 1024.

Define environment as TZ=Etc/UT

POD_NAME itsvalueFrom:

fieldRe:

fieldPath: metadata.name

POD_NAMESPACE its

valueFrom:

fieldRe:

fieldPath: metadata.namespace

```
lab5 > ! haproxy-ingress.yml > {} spec > {} template > {} spec > [ ] containers > {} 0 > [ ] env
io.k8s.api.apps.v1.Deployment (v1@deployment.json)
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    name: haproxy-ingress-devops
5    namespace: haproxy-controller-devops
6    labels:
7      run: haproxy-ingress
8  spec:
9    replicas: 1
10   selector:
11     matchLabels:
12       run: haproxy-ingress
13   template:
14     metadata:
15       labels:
16         run: haproxy-ingress
17     spec:
18       serviceAccountName: haproxy-service-account-devops
19       containers:
20         - name: ingress-container-devops
21           image: haproxytech/kubernetes-ingress
22           args:
23             - "--default-backend-service=haproxy-controller-devops/service-backend-devops"
24           ports:
25             - name: http
26               containerPort: 80
27             - name: https
28               containerPort: 443
29             - name: stat
30               containerPort: 1024
31           resources:
32             requests:
33               cpu: "500m"
```

```

lab5 > ! haproxy-ingress.yml > {} spec > {} template > {} spec > [ ] containers > {} 0 > [ ] env
8   spec:
13   template:
17     spec:
19       containers:
20         - name: ingress-container-devops
21           containerPort: 1024
31           resources:
32             requests:
33               cpu: "500m"
34               memory: "50Mi"
35           env:
36             - name: TZ
37               value: Etc/UTC
38             - name: POD_NAME
39               valueFrom:
40                 fieldRef:
41                   fieldPath: metadata.name
42             - name: POD_NAMESPACE
43               valueFrom:
44                 fieldRef:
45                   fieldPath: metadata.namespace
46           livenessProbe:
47             httpGet:
48               path: /health
49               port: 1024
50             initialDelaySeconds: 3
51             periodSeconds: 3
52

```

8. Create a service for frontend which should be named as ingress-service-devops under same namespace, type should be NodePort. The first port name should be http, its port should be 80, protocol should be TCP, targetPort should be 80 and nodePort should be 32456. The second port name should be https, its port should be 443, protocol should be TCP, targetPort should be 443 and nodePort should be 32567. The third port name should be stat, its port should be 1024, protocol should be TCP, targetPort should be 1024 and nodePort should be 32678.

9. Access the proxy states on port “1024” via the nodePort service “ingress-service-devops” via the browser

HAProxy version 3.0.7-ce35390, released 2024/12/12
Statistics Report for pid 121

> General process information

```
pid = 121 (process fit, nproc= 1, rthruend= 2)
uptime = 0d 00m02s, usings= 0
system memory: memmax = 2461 MB, ulimit = 142149
maxsock = 142149, maxconns = 72000, reached = 0, maxpipes = 0
current cores = 1, current pipes = 0, core size = 0K, bit rate = 0 Kbps
Running tasks: 0f0e (0 mced); idle % 100 %
```

- active UP
- active UP going down
- active DOWN, going up
- active or backup DOWN
- active or backup DOWN for maintenance (MAINT)
- active or backup SOFT STOPPED for maintenance
- Note: "NOLE" DRAIN" = UP with load-balancing disabled.

- backup UP
- backup UP going down
- backup DOWN, going up
- [not checked]
- active or backup DOWN for maintenance (MAINT)
- active or backup SOFT STOPPED for maintenance

Display options: Scope:

- Hide DOWN servers
- Disable info
- Refresh
- Csv export
- CSV report (schema)

External resources:

- Privacy site
- Updates v3.0
- Online manual

Process		Queue				Session rate				Sessions				Bytes				Denied				Errors				Warnings				Status				Server			
Frontend	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Sessions	Total	LbTot	Last	In	Out	Rsp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Downtime	Thrtle							
Frontend	0	0	0	-	0	0	0	0	0	72 000	0			0	0	0	0	0	0	0	0	OPEN															

Info		Queue				Session rate				Sessions				Bytes				Denied				Errors				Warnings				Status				Server			
Frontend	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Sessions	Total	LbTot	Last	In	Out	Rsp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Downtime	Thrtle							
Frontend	0	0	0	-	0	0	0	0	0	72 000	0			0	0	0	0	0	0	0	0	OPEN															

Stats		Queue				Session rate				Sessions				Bytes				Denied				Errors				Warnings				Status				Server			
Frontend	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Sessions	Total	LbTot	Last	In	Out	Rsp	Req	Conn	Resp	Retr	Redis	Status	LastChk	Wght	Act	Bck	Chk	Dwn	Downtime	Thrtle							
Frontend	0	0	0	-	0	0	0	0	0	72 000	0			1 510	616 602	0	0	0	0	0	0	OPEN															

haproxy controller displays anemethash: info

		Queue				Session rate				Sessions				Bytes				Denied				Errors				Warnings				Status				Server			
	Cur	Max	Limit	Cur	Max	Limit	Cur	Max	Limit	Sessions	Total	LbTot	Last	In	Out	Rsp	Req	Conn	Resp	Retr	Redis		LastChk	Wght	Act	Bck	Chk	Dwn	Downtime	Thrtle							
SRV_1	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s UP	LOCK in Drn	1/1	Y	-	0	0	0	23%	-						
SRV_2	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s MAINT		1/1	Y	-	0	0	0	23%	-						
SRV_3	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s MAINT		1/1	Y	-	0	0	0	23%	-						
SRV_4	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s MAINT		1/1	Y	-	0	0	0	23%	-						
SRV_5	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s MAINT		1/1	Y	-	0	0	0	23%	-						
SRV_6	0	0	0	-	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	23s MAINT		1/1	Y	-	0	0	0	23%	-						