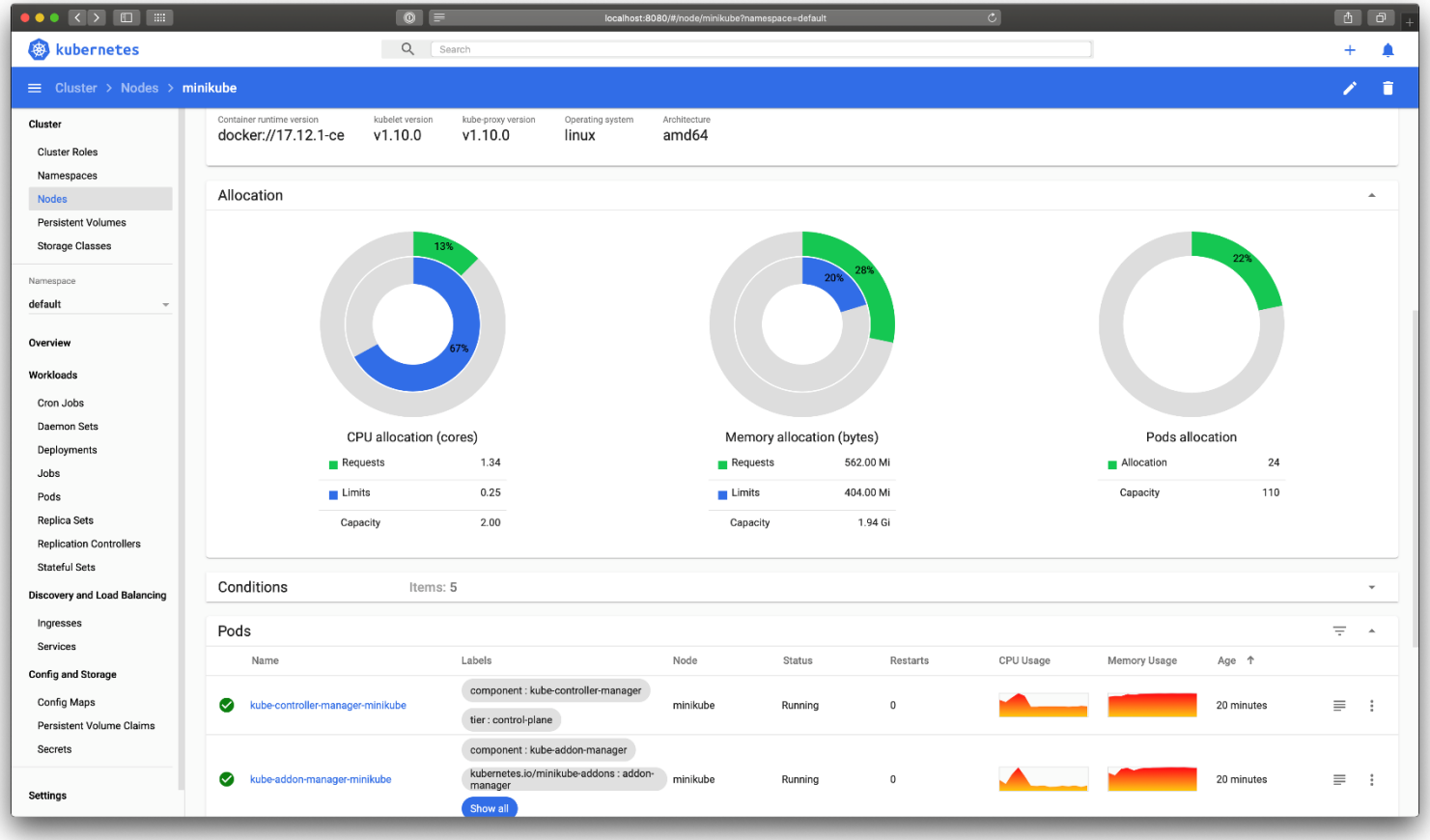
Kubernetes Dashboard

Kubernetes Dashboard is a general purpose, web-based UI for Kubernetes clusters. It allows users to manage applications running in the cluster and troubleshoot them, as well as manage the cluster itself.



Getting Started

IMPORTANT: Read the Access Control guide before performing any further steps. The default Dashboard deployment contains a minimal set of RBAC privileges needed to run.

To deploy Dashboard, execute following command:

```
$ kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v1.10.1/src/deploy/recommended/kubernetes-dashboard.yaml
```

To access Dashboard from your local workstation you must create a secure channel to your Kubernetes cluster. Run the following command:

```
$ kubectl proxy
```

Now access Dashboard at:

http://localhost:8001/api/v1/namespaces/kube-system/services/https:kubernetes-dashboard:/proxy/.

Create An Authentication Token (RBAC)

To find out how to create sample user and log in follow Creating sample user guide.

NOTE:

- Kubeconfig Authentication method does not support external identity providers or certificate-based authentication.
- Dashboard can only be accessed over HTTPS
- Heapster has to be running in the cluster for the metrics and graphs to be available. Read more about it in Integrationsguide.

Documentation

Dashboard documentation can be found on Wiki pages which contains:

- Common: Entry-level overview
- User Guide: Installation, Accessing Dashboard and more for users
- Developer Guide: Getting Started, Dependency Management and more for anyone interested in contributing

Community, discussion, contribution, and support

Learn how to engage with the Kubernetes community on the community page.

You can reach the maintainers of this project at:

- **#sig-ui on Kubernetes Slack**
- **kubernetes-sig-ui mailing list**
- **Issue tracker**
- **SIG info**

Code of conduct

Participation in the Kubernetes community is governed by the Kubernetes Code of Conduct.

License

Ref: https://github.com/kubernetes/dashboard

==========

Creating sample user

In this guide, we will find out how to create a new user using Service Account mechanism of Kubernetes, grant this user admin permissions and log in to Dashboard using bearer token tied to this user.

Copy provided snippets to some `dashboard-adminuser.yaml` file and use `kubectl apply -f dashboard-adminuser.yaml` to create them.

Create Service Account

We are creating Service Account with name `admin-user` in namespace `kube-system` first.

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: admin-user
  namespace: kube-system
```

Create ClusterRoleBinding

In most cases after provisioning our cluster using `kops` or `kubeadm` or any other popular tool, the `ClusterRole` `admin-Role` already exists in the cluster. We can use it and create only `ClusterRoleBinding` for our `ServiceAccount`.
NOTE: `apiVersion` of `ClusterRoleBinding` resource may differ between Kubernetes versions. Prior to Kubernetes `v1.8` the `apiVersion` was `rbac.authorization.k8s.io/v1beta1`.

```yaml
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRoleBinding
metadata:
  name: admin-user
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
```

```
  name: cluster-admin
subjects:
- kind: ServiceAccount
  name: admin-user
  namespace: kube-system
```

Bearer Token

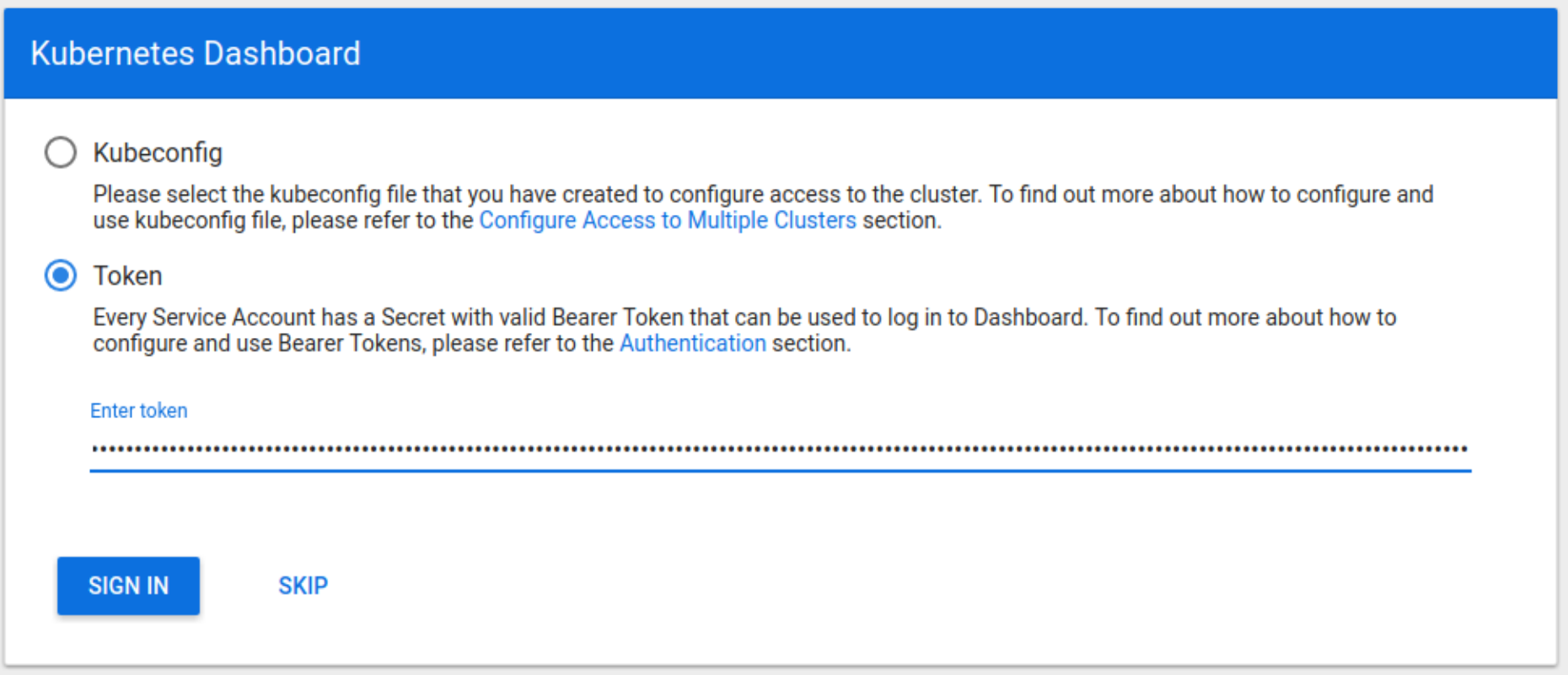
Now we need to find token we can use to log in. Execute following command:

```
kubectl -n kube-system describe secret $(kubectl -n kube-system get secret | grep admin-user | awk '{print $1}')
```

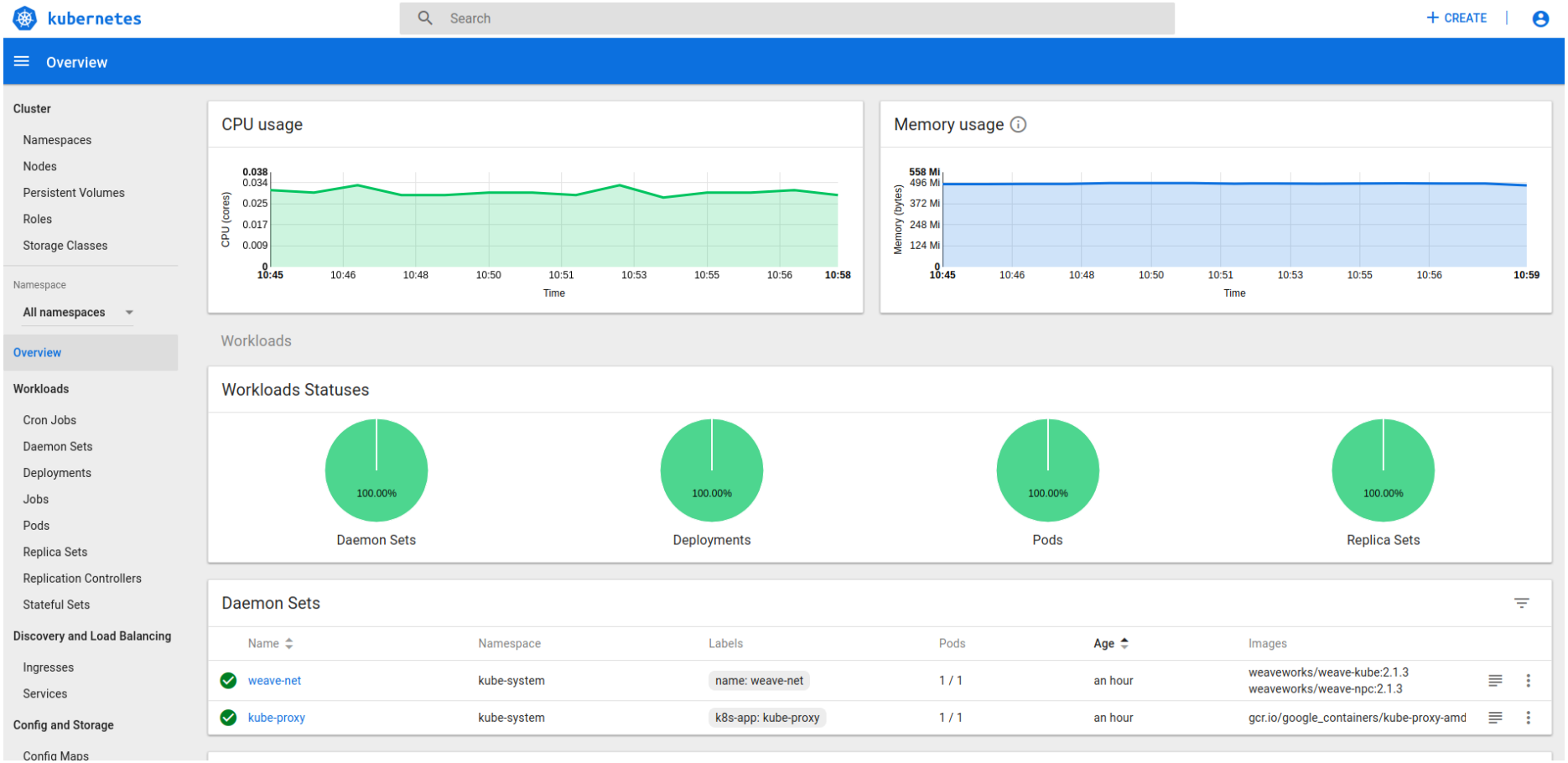
It should print something like:

```
Name:         admin-user-token-6gl6l
Namespace:    kube-system
Labels:       <none>
Annotations:  kubernetes.io/service-account.name=admin-user
              kubernetes.io/service-account.uid=b16afba9-dfec-11e7-bbb9-901b0e532516

Type:  kubernetes.io/service-account-token

Data
====
ca.crt:     1025 bytes
namespace:  11 bytes
token:
eyJhbGciOiJSUzI1NiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJrdWJlcm5ldGVzL3NlcnZpY2VhY2NvdW50Iiwia3ViZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9uY
W1lc3BhY2UiOiJrdWJlLXN5c3RlbSIsImt1YmVybmV0ZXMuaW8vc2VydmljZWFjY291bnQvc2VjcmV0Lm5hbWUiOiJhZG1pbi11c2VyLXRva2VuLTZnbDZsIiwia3V
iZXJuZXRlcy5pby9zZXJ2aWNlYWNjb3VudC9zZXJ2aWNlLWFjY291bnQubmFtZSI6ImFkbWluLXVzZXIiLCJrdWJlcm5ldGVzLmlvL3NlcnZpY2VhY2NvdW50L3Nlc
nZpY2UtYWNjb3VudC51aWQiOiJiMTZhZmJhOS1kZmVjLTExZTctYmJiOS05MDFiMGU1MzI1MTYiLCJzdWIiOiJzeXN0ZW06c2VydmljZWFjY291bnQ6a3ViZS1zeXN
0ZW06YWRtaW4tdXNlciJ9.M70CU3lbu3PP4OjhFms8PVL5pQKj-jj4RNSLA4YmQfTXpPUuxqXjiTf094_Rzr0fgN_IVX6gC4fiNUL5ynx9KU-
lkPfk0HnX8scxfJNzypL039mpGt0bbe1IXKSIRaq_9VW59Xz-
yBUhycYcKPO9RM2Qa1Ax29nqNVko4vLn1_1wPqJ6XSq3GYI8anTzV8Fku4jasUwjrws6Cn6_sPEGmL54sq5R4Z5afUtv-
mItTmqZZdxnkRqcJLlg2Y8WbCPogErbsaCDJoABQ7ppaqHetwfM_0yMun6ABOQbIwwl8pspJhpplKwyo700OSpvTT9zlBsu-b35lzXGBRHzv5g_RA
```

Now copy the token and paste it into `Enter token` field on log in screen.



Click `Sign in` button and that's it. You are now logged in as an admin.



In order to find out more about how to grant/deny permissions in Kubernetes read official authentication & authorization documentation.

Ref: https://github.com/kubernetes/dashboard/wiki/Creating-sample-user