



Kube2SONiC

Container Management

Oct 2023

Revision History

Revision No.	Description	Editor	Date
1.0	Document Creation	Saba Akram	Oct 06, 2023

Table of Contents

Introduction	3
K8s Cluster	4
Verify Installation	4
Container management	4
Deploy k8s Dashboard	4

Introduction

A Kubernetes dashboard is a web-based user interface designed for managing Kubernetes clusters. It serves as a crucial tool for deploying and managing containerized applications to Kubernetes clusters, monitoring and troubleshooting those applications, and overseeing the management of the cluster and its associated resources. This user-friendly interface streamlines the interaction with Kubernetes, facilitating efficient container orchestration and cluster administration tasks.

K8s Cluster

Node	OS version
Master	Ubuntu 22.0
Worker	SONiC

The kubernetes cluster should be deployed with at least one master and one worker node.

Verify Installation

Check and verify all nodes are in Ready state

```
Python
kubectl get nodes
```

Check and verify all the k8s pods are up and running

```
Python
kubectl get pods -n kube-system
```

Container management

Deploy k8s Dashboard

Run the following command to deploy the dashboard:

```
Python
kubectl apply -f https://raw.githubusercontent.com/kubernetes/dashboard/v2.7.0/aio/deploy/recommended.yaml
```

Check and verify all pods of **kubernetes-dashboard** are running

```
Python
kubectl get pods -n kubernetes-dashboard
```

Access the dashboard

Start the proxy server between your machine and Kubernetes API server.

```
Python
kubectl proxy
```

Access the dashboard from browser and enter the URL

<http://localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kubernetes-dashboard:/proxy/#/login>

Create Admin Service Account

```
Python
kubectl create serviceaccount dashboard -n default
```

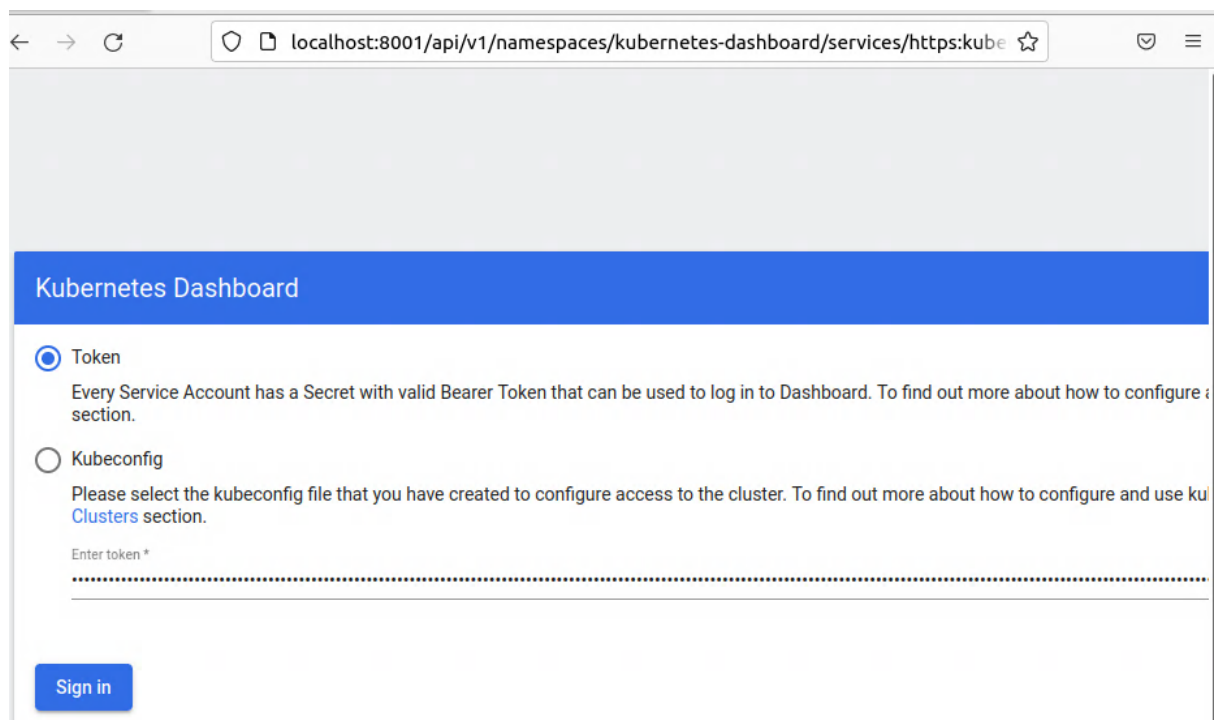
Add the cluster role binding

```
Python
kubectl create clusterrolebinding dashboard-admin -n default
--clusterrole=cluster-admin --serviceaccount=default:dashboard
```

Get the secret token to access the dashboard

```
Python
kubectl get secret $(kubectl get serviceaccount dashboard -o
jsonpath="{.secrets[0].name}") -o jsonpath="{.data.token}" | base64 --decode
```

Copy the secret token and paste it in Dashboard Login Page, by selecting a token option



← → ↺ localhost:8001/api/v1/namespaces/kubernetes-dashboard/services/https:kube

Kubernetes Dashboard

☒ Token
Every Service Account has a Secret with valid Bearer Token that can be used to log in to Dashboard. To find out more about how to configure section.

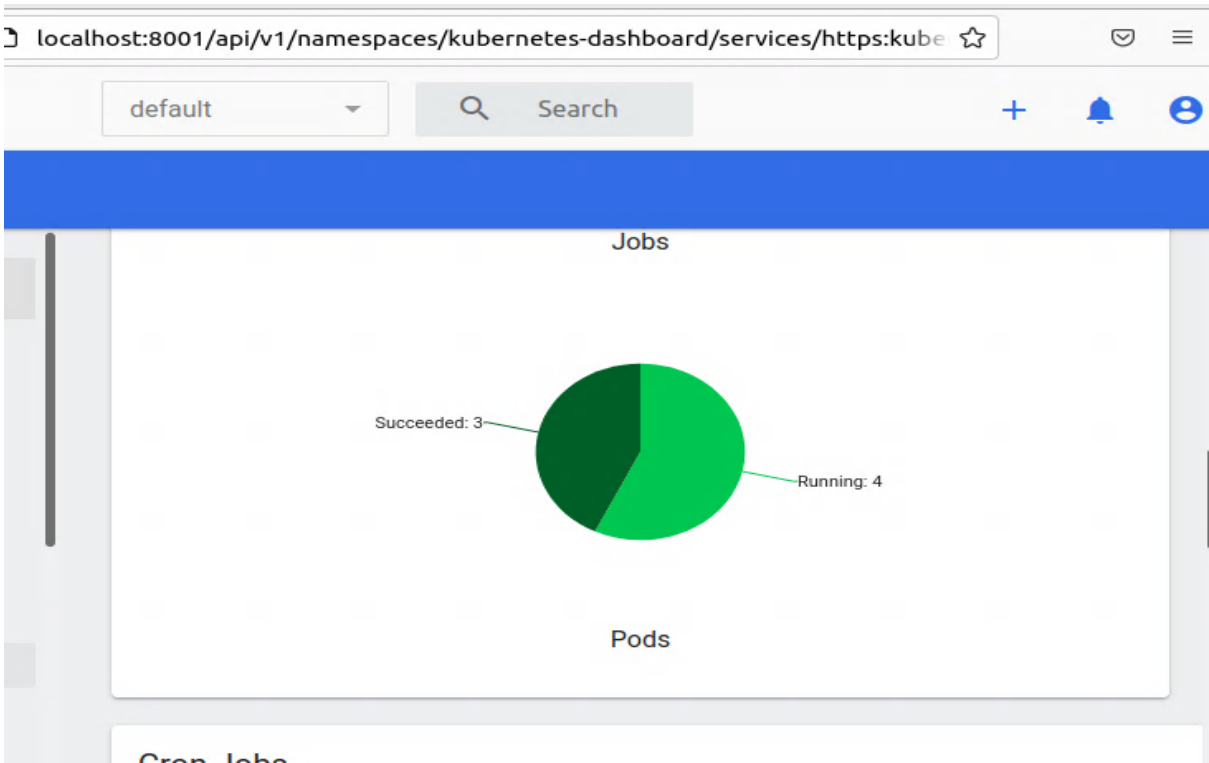
☐ Kubeconfig
Please select the kubeconfig file that you have created to configure access to the cluster. To find out more about how to configure and use ku Clusters section.

Enter token *

.....

Sign in

Access pods



Now you can manage your SONiC containers through kubernetes dashboard

	snmp-ctjpb	docker-snmp:latest	controller-revision-hash: d f4bf78dc	name: snmp	worker1	Running	0	-	-
			pod-template-generation: 1						
	sonic-mgmt-xtwx	docker-sonic-mgmt-frame work:latest	controller-revision-hash: 8 cb769b4d	name: sonic-mgmt	worker2	Running	0	-	-
			pod-template-generation: 1						
	pmon-xnnxh	docker-sonic-telemetry:lat est	controller-revision-hash: c bc69b866	name: pmon	worker2	Running	0	-	-
			pod-template-generation: 1						
	telemetry-8rff2	docker-sonic-telemetry:lat est	controller-revision-hash: 5 6d5f78999	name: telemetry	worker2	Running	0	-	-
			pod-template-generation: 1						

You can access the shell and execute commands

```
Shell in snmp      in snmp-ctjpp

root@worker1:~# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group default qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
        valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast state UP group default qlen 1000
    link/ether 52:54:00:68:0d:90 brd ff:ff:ff:ff:ff:ff
    inet 192.168.122.12/24 brd 192.168.122.255 scope global dynamic eth0
        valid_lft 2961sec preferred_lft 2961sec
    inet6 fe80::5054:ff:fe68:d90/64 scope link
        valid_lft forever preferred_lft forever
3: docker0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc noqueue state DOWN group default
    link/ether 02:42:28:e7:8b:f2 brd ff:ff:ff:ff:ff:ff
    inet 240.127.1.1/24 brd 240.127.1.255 scope global docker0
        valid_lft forever preferred_lft forever
    inet6 fd00::1/80 scope global
        valid_lft forever preferred_lft forever
    inet6 fe80::1/64 scope link
        valid_lft forever preferred_lft forever
4: Ethernet24: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9100 qdisc mq state UNKNOWN group default qlen 1000
    link/ether 52:54:00:ee:91:3a brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.12/31 scope global Ethernet24
        valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:feee:913a/64 scope link
        valid_lft forever preferred_lft forever
5: Ethernet28: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 9100 qdisc mq state UNKNOWN group default qlen 1000
    link/ether 52:54:00:ee:91:3a brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.14/31 scope global Ethernet28
        valid_lft forever preferred_lft forever
    inet6 fe80::5054:ff:feee:913a/64 scope link
        valid_lft forever preferred_lft forever
```

Workloads > Pods > snmp-ctjpp

Workloads

Cron Jobs

Daemon Sets

Deployments

Jobs

Pods

Replica Sets

Replication Controllers

Stateful Sets

Service

Ingresses

Services

Config and Storage

Config Maps

Persistent Volume Claims

Secrets

Storage Classes

Cluster

Cluster Role Bindings

Cluster Roles

Namespaces

Metadata

Name	Namespace	Created	Age	UID
snmp-ctjpp	default	Oct 5, 2023	a day ago	ad849fb8-a504-4379-a9e0-45ad9783f0ad

Labels

controller-revision-hash: df4bf78dcname: snmppod-template-generation: 1

Resource information

Node	Status	IP	QoS Class	Restarts	Service Account
worker1	Running	192.168.122.12	BestEffort	0	default

Conditions

Type	Status	Last probe time	Last transition time	Reason
Initialized	True	..	a day ago	-
Ready	True	..	a day ago	-
ContainersReady	True	..	a day ago	-
PodScheduled	True	..	a day ago	-