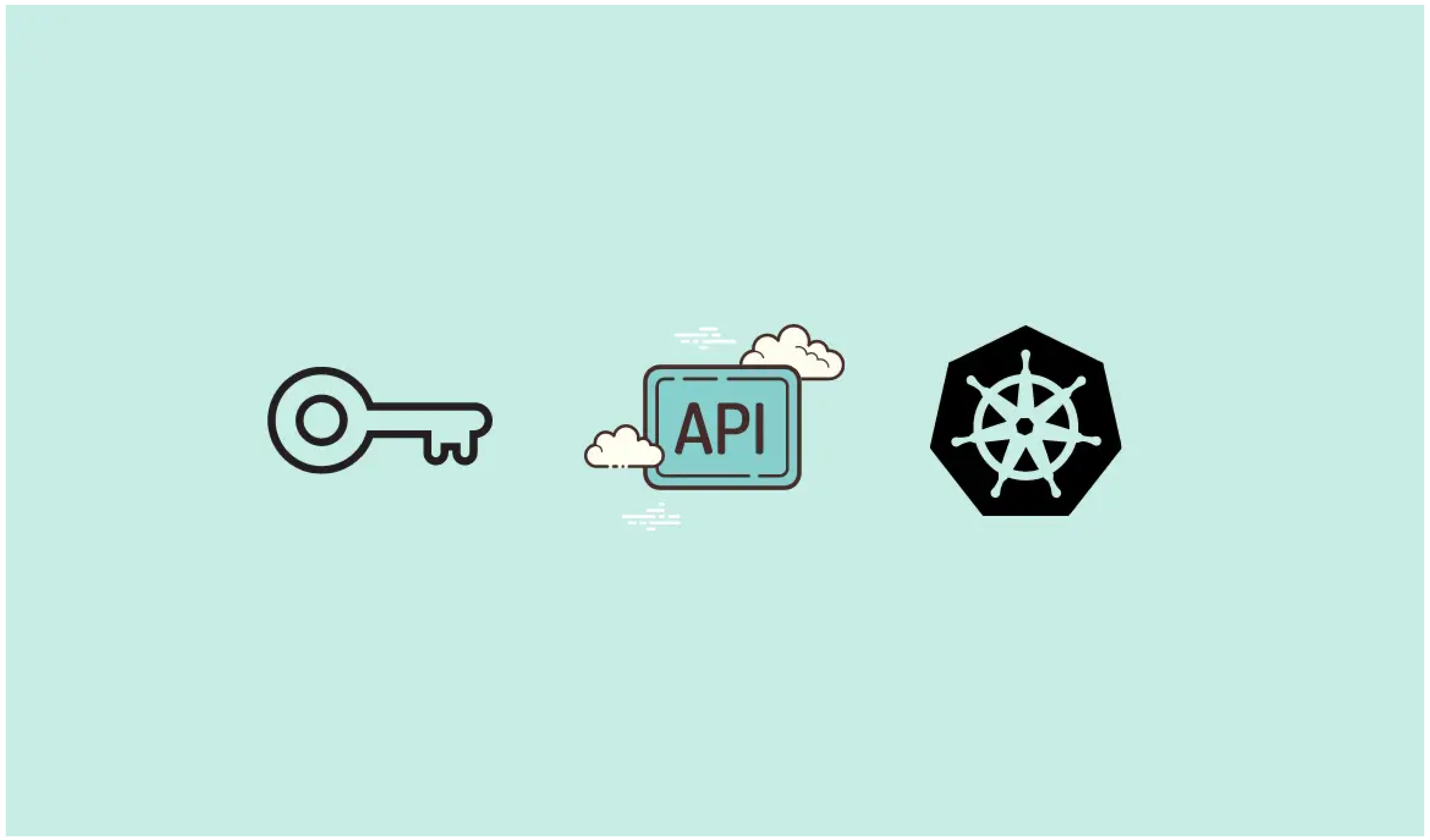
— KUBERNETES

How to Create kubernetes Role for Service Account

by **Bibin Wilson** · June 1, 2021



In this blog, you will learn how to create Kubernetes role for a service account and use it with the pods, deployments, and cronjobs.

> **Note:** A `role` provides API access only to resources present in a namespace. For cluster-wide API access, you should use a `ClusterRole`

Create Kubernetes Role for Service Account

Let’s consider the following scenario

1. You have deployments/pods in a namespace called `webapps`
2. The deployments/pods need Kubernetes API access to manage resources in a namespace.

The solution to the above scenarios is to have a service account with roles with specific API access.

1. Create a service account bound to the namespace webapps namespace
2. Create a role with the list of required API access to Kubernetes resources.
3. Create a Rolebinding to bind the role to the service account.
4. Use the service account in the pod/deployment or Kubernetes Cronjobs

Lets implement it.

Create webapps Namespace

For the purpose of demonstration, we will create a namespace called `webapps`

```
kubectl create namespace webapps
```

Create Kubernetes Service Account

Let’s create a service account named `app-service-account` that bounds to `webapps` namespace

Copy the following and execute directly on the terminal.

```
cat <<EOF | kubectl apply -f -
apiVersion: v1
kind: ServiceAccount
metadata:
  name: app-service-account
  namespace: webapps
EOF
```

Create a Role For API Access

In the kubernetes Role, we specify the list of API access required for Kubernetes resources.

Note: The following role has access to most Kubernetes resources with all read, write, list, update, patch, and delete permissions. When you implement it in real projects, you should add only the required resources and actions to the role.

Lets create a role named `app-role` specific to `webapps` namespace.

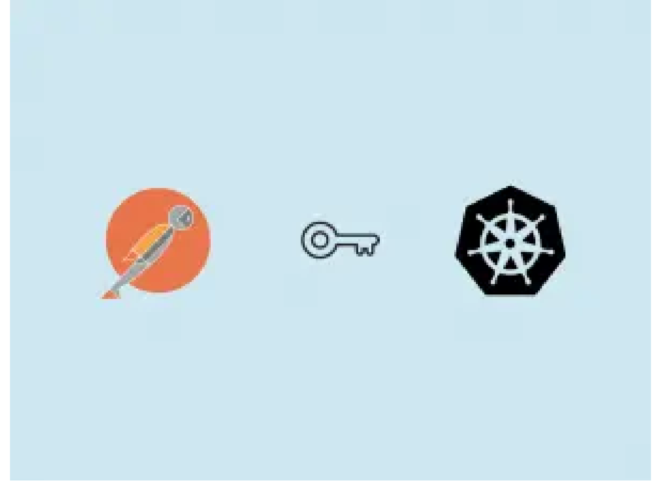
Copy the following and execute directly on the terminal.

```
cat <<EOF | kubectl apply -f -
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: app-role
  namespace: webapps
rules:
  - apiGroups:
        - ""
        - apps
        - autoscaling
        - batch
        - extensions
        - policy
        - rbac.authorization.k8s.io
    resources:
      - pods
      - componentstatuses
      - configmaps
      - daemonsets
      - deployments
      - events
      - endpoints
      - horizontalpodautoscalers
      - ingress
      - jobs
      - limitranges
      - namespaces
      - nodes
      - pods
      - persistentvolumes
      - persistentvolumeclaims
      - resourcequotas
      - replicasets
      - replicationcontrollers
      - serviceaccounts
      - services
    verbs: ["get", "list", "watch", "create", "update", "patch", "delete"]
EOF
```

Lets list the role.

```
kubectl get roles -n webapps
```

ALSO READ



— KUBERNETES

How To Create Kubernetes Service Account For API Access

by Bibin Wilson · June 5, 2021

Create a Rolebinding [Attaching Role to ServiceAccount]

Now we have a service account and a role which has no relation.

With Rolebinding we attach the role to the service account. So the pods which use the service account in `webapps` namespace will have all the access mentioned in the `app-role`

Copy the following and execute directly on the terminal.

```
cat <<EOF | kubectl apply -f -
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: app-rolebinding
  namespace: webapps
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: Role
  name: app-role
subjects:
- namespace: webapps
  kind: ServiceAccount
  name: app-service-account
EOF
```

Validate Kubernetes Role Permissions

We will use the `bibinwilson/docker-kubectl` Docker image that I have created with the kubectl utility.

Let's deploy a pod named `debug` with bibinwilson/docker-kubectl image and our service account `app-service-account`.

```
cat <<EOF | kubectl apply -f -
---
apiVersion: v1
kind: Pod
metadata:
  name: debug
  namespace: webapps
```

```
spec:
  containers:
  - image: bibinwilson/docker-kubectl:latest
    name: kubectl
  serviceAccountName: app-service-account
EOF
```

Lets `exec` in to the `debug` pod and see if has the privileges we mentioned in the role.

```
kubectl exec -it debug /bin/bash -n webapps
```

Now, you should be able to list pods and other resources in `webapps` namespace. You cannot list the pods in other namespaces are this role is specific to `webapps` namespace.

If you deploy a pod without the service account and list the pods, you will get the following error.

```
Error from server (Forbidden): pods is forbidden: User
"system:serviceaccount:webapps:default" cannot list resource "pods" in API group
"" in the namespace "webapps"
```

The default service account that gets attached to pods doesn't have any API access to resources.

Using Service Account with Kubernetes Cronjob

Here is an example of Kubernetes Cronjob with a service account.

```
apiVersion: batch/v1beta1
kind: CronJob
metadata:
    name: kubernetes-cron-job
spec:
  schedule: "0,15,30,45 * * * *"
  jobTemplate:
    spec:
      template:
        metadata:
          labels:
            app: cron-batch-job
        spec:
          restartPolicy: OnFailure
          serviceAccountName: app-service-account
          containers:
          - name: kube-cron-job
            image: devopscube/kubernetes-job-demo:latest
            args: ["100"]
```

Using Service Account With Kubernetes Deployment

Here is an example of a Kubernetes deployment with a service account.

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-deployment
  labels:
    app: nginx
spec:
  replicas: 3
  selector:
    matchLabels:
      app: nginx
  template:
    metadata:
      labels:
        app: nginx
    spec:
      serviceAccountName: app-service-account
      containers:
      - name: nginx
        image: nginx:1.14.2
        ports:
        - containerPort: 80
```

Conclusion

In this blog post, I have added all the steps required to create Kubernetes role and use it with the pod, deployment, and Cronjonbs.

There are particularly not many use cases where you need the namespace specific roles.

One main use would be for creating users with access limited to a namespace. Also, to create service accounts to have API access to namespaces from external applications.

Let me know if you face any issues or have any questions related to Kubernetes roles.

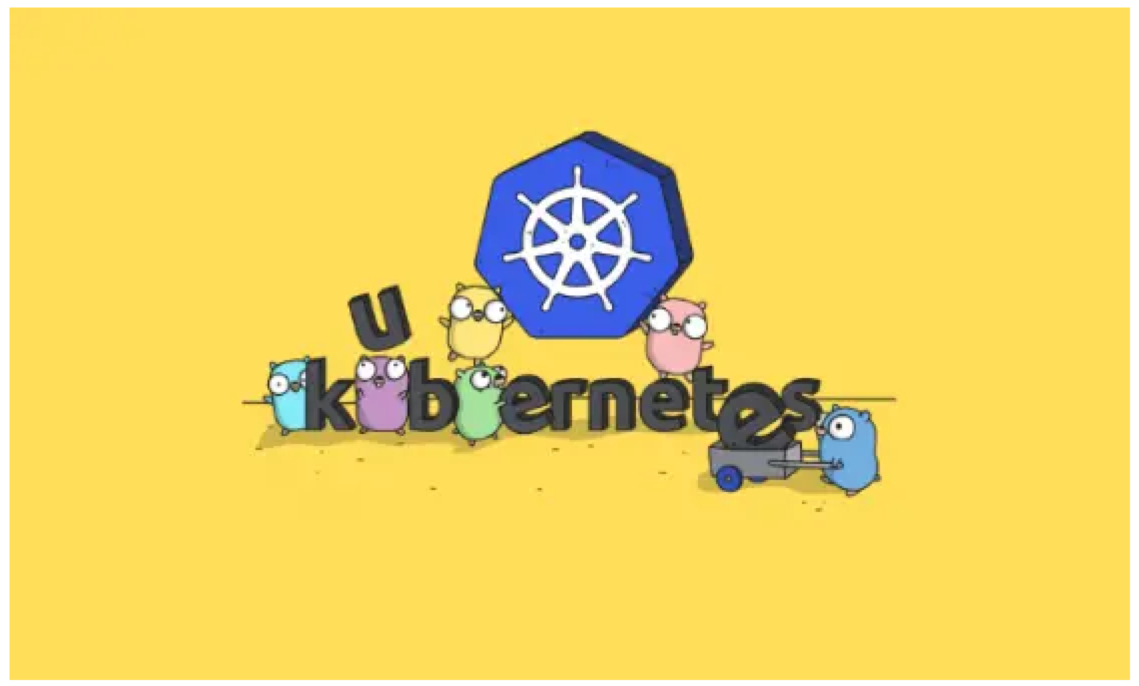
SHARE TWEET

Bibin Wilson

An author, blogger and DevOps practitioner. In spare time, he loves to try out the latest open source technologies. He works as an Associate Technical Architect

VIEW COMMENTS (0)

YOU MAY ALSO LIKE

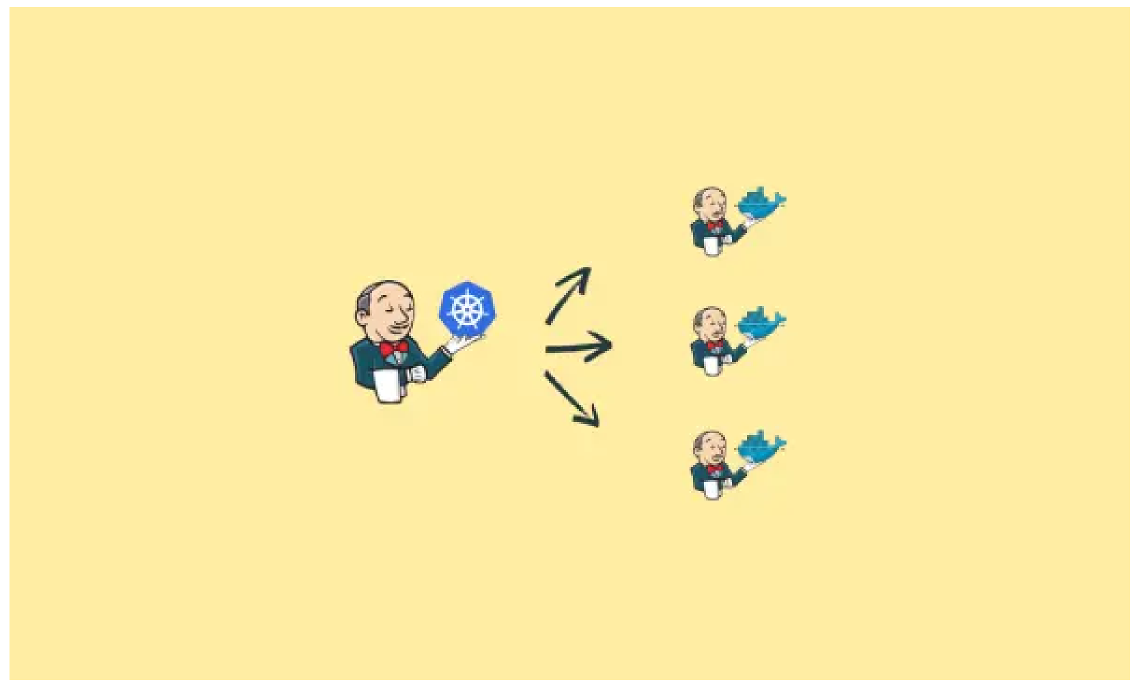


C — COMMON

Kubernetes Deployment Tutorial For Beginners

by **devopscube** · November 29, 2018

This Kubernetes deployment tutorial guide will explain the key concepts in a Kubernetes YAML specification with an Nginx...

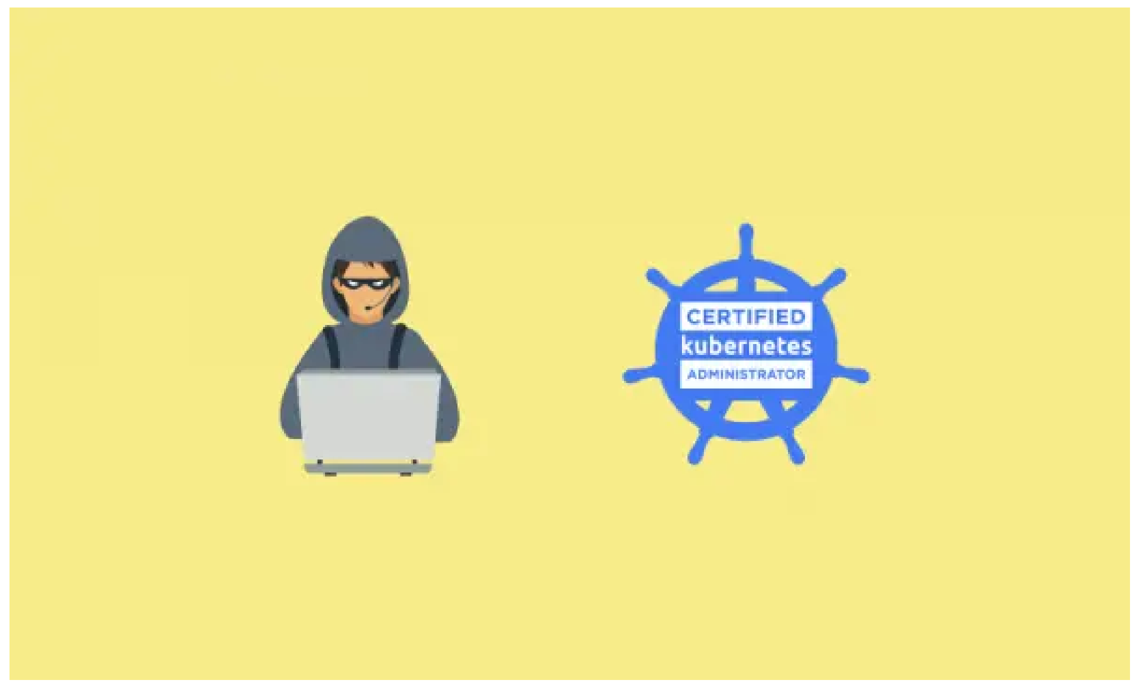


J — JENKINS

How to Setup Jenkins Build Agents on Kubernetes Cluster

by **Bibin Wilson** · July 4, 2021

In this Jenkins tutorial, I explained the detailed steps to set up Jenkins master and scale Jenkins build...



K — KUBERNETES

CKA Exam Study Guide: A Complete Resource For CKA Aspirants

by **Shishir Khandelwal** · June 25, 2021

This CKA Exam study guide will help you prepare for the Certified Kubernetes Administrator (CKA) exam with all...

T — TUTORIAL

Kubernetes Tutorials For Beginners: Getting Started Guide

by **devopscube** · April 14, 2020

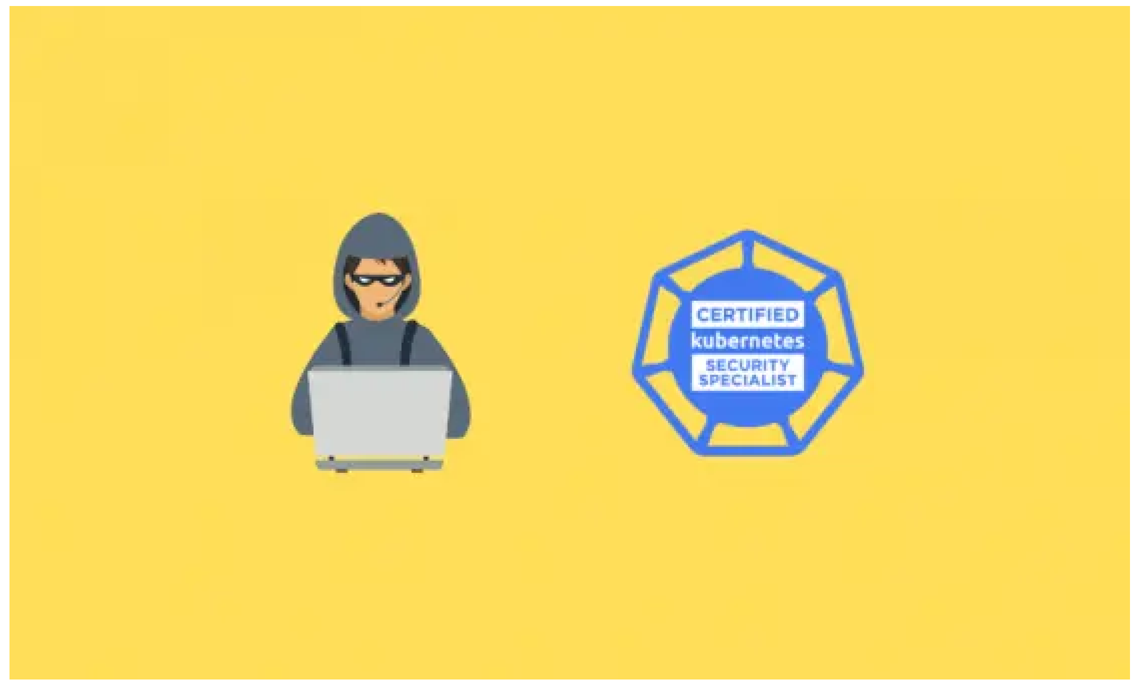
Official kubernetes (k8s) website says, Kubernetes is an open-source system for automating deployment, scaling, and management of containerized applications....

D — DEVOPS

How to Setup Prometheus Node Exporter on Kubernetes

by **devopscube** · April 6, 2021

If you want to know how the Kubernetes nodes perform or monitor system-level insights of kubernetes nodes, you...



K — KUBERNETES

CKS Exam Study Guide: Resources to Pass Certified Kubernetes Security Specialist

by **Bibin Wilson** · June 23, 2021

In this Certified Kubernetes Security Specialist (CKS) Exam study guide, I have listed all the resources you can...