1. Verify the Service Account Token is Mounted

When a pod is associated with a service account, Kubernetes mounts the service account's token and certificate in the pod.

Command:

```
kubectl exec -it my-pod -- ls
/var/run/secrets/kubernetes.io/serviceaccount/
```

Expected Output:

You should see files like:

```
ca.crt
namespace
token
```

The `token` file is the service account token used by the pod to authenticate with the Kubernetes API.

Check the Token Contents:

```
kubectl exec -it my-pod -- cat
/var/run/secrets/kubernetes.io/serviceaccount/token
```

This will display the token that the pod can use to interact with the Kubernetes API.

---

2. Use the Token to Access the Kubernetes API

You can manually test if the token allows access to the Kubernetes API.

Fetch the Kubernetes API Server URL:

```
kubectl config view --minify -o
jsonpath='{.clusters[0].cluster.server}'
```

Test Access from Inside the Pod:

Start a shell inside the pod:

```
kubectl exec -it my-pod -- sh
```

Use `curl` with the service account token to list pods:

```
TOKEN=$(cat /var/run/secrets/kubernetes.io/serviceaccount/token)
curl -s --header "Authorization: Bearer $TOKEN" --cacert
/var/run/secrets/kubernetes.io/serviceaccount/ca.crt
https://<KUBERNETES_API_SERVER>/api/v1/namespaces/default/pods
```

Replace `<KUBERNETES_API_SERVER>` with the URL you obtained earlier.

Expected Output:

If the service account and RBAC are correctly configured, you'll see a list of pods in the `default` namespace in JSON format.

---

3. Check the Pod's Logs

If your application inside the pod interacts with the Kubernetes API, check the pod's logs to ensure it can perform the intended actions.

Command:

```
kubectl logs my-pod
```

Look for any errors or successful API interactions in the logs.

---

4. Simulate Access with `kubectl` and the Service Account Token

You can also test the service account's permissions outside the pod by using its token directly with `kubectl`.

Get the Token:

```
kubectl get secret $(kubectl get serviceaccount
my-custom-service-account -o jsonpath='{.secrets[0].name}') -o
jsonpath='{.data.token}' | base64 --decode
```

Test Access:

Run a `kubectl` command using the token:

```
kubectl --token=<TOKEN> --server=$(kubectl config view --minify -o
jsonpath='{.clusters[0].cluster.server}') --insecure-skip-tls-verify
get pods
```

Expected Output:

If the service account is properly configured, you'll see the list of pods (or whatever permissions the service account has been granted).

---

5. Debugging

If something isn't working:

Check the Pod's Service Account:

```
kubectl get pod my-pod -o jsonpath='{.spec.serviceAccountName}'
```

Review RBAC Permissions:

```
kubectl describe rolebinding read-pods-binding
kubectl describe role pod-reader
```

Check for Errors in Logs:

```
kubectl logs my-pod
```