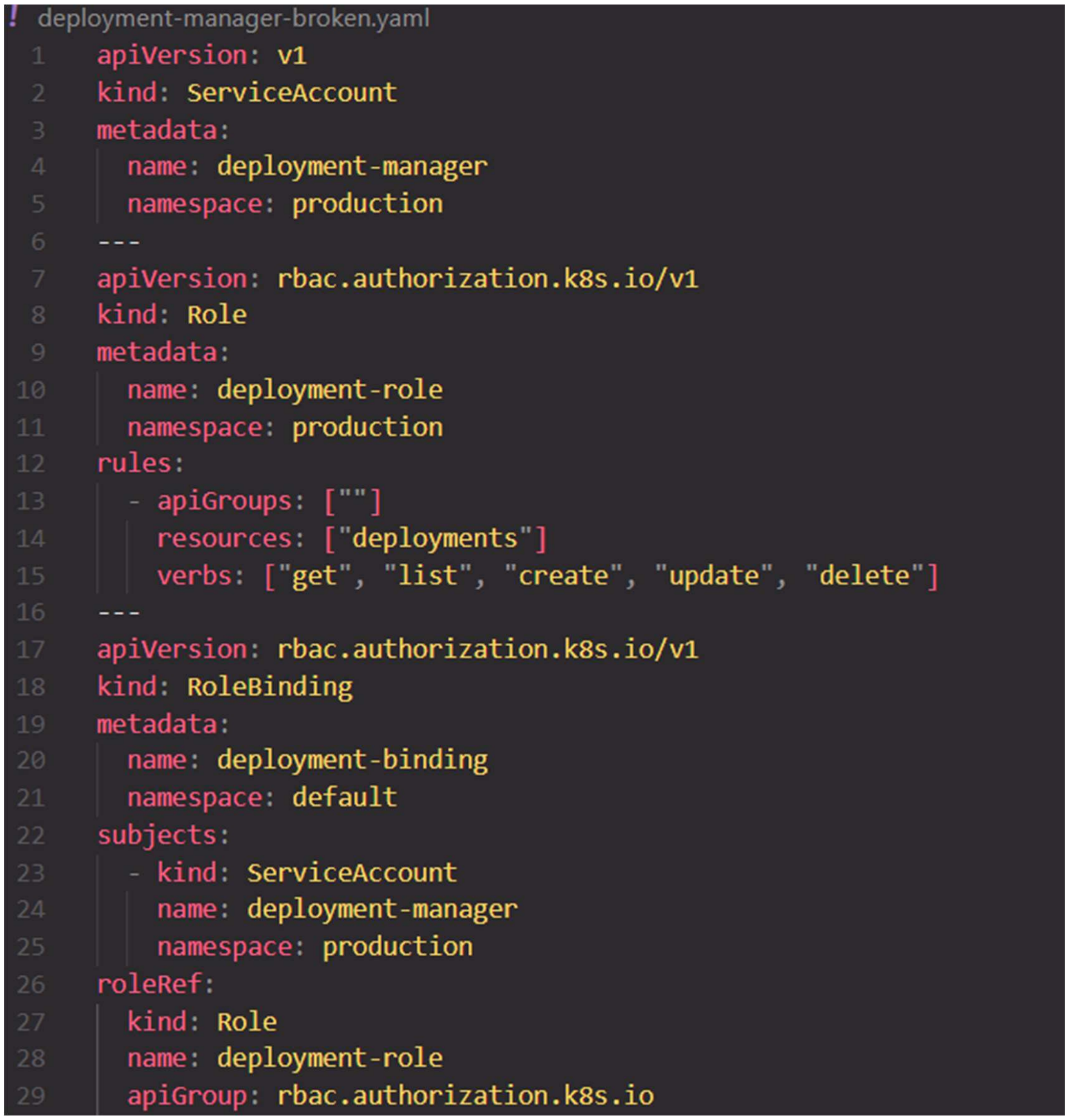
Part-4: Troubleshooting RBAC Issues

Problem Statement

A service account deployment manager in the production namespace should be able to manage deployments but keeps getting permission errors.

Given Broken Configuration Screenshot



deployment-manager-broken.yaml

```yaml
apiVersion: v1
kind: ServiceAccount
metadata:
  name: deployment-manager
  namespace: production
---
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: deployment-role
  namespace: production
rules:
  - apiGroups: [""]
    resources: ["deployments"]
    verbs: ["get", "list", "create", "update", "delete"]
---
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: deployment-binding
  namespace: default
subjects:
  - kind: ServiceAccount
    name: deployment-manager
    namespace: production
roleRef:
  kind: Role
  name: deployment-role
  apiGroup: rbac.authorization.k8s.io
```

Troubleshooting Process

Step 1: Analyze the Configuration

Reviewed all three resources:

- ServiceAccount
- Role
- RoleBinding

Step 2: Identify Errors

Error #1: Incorrect API Group

Location: Role rules section

Finding: apiGroups: [""]

Problem: Deployments belong to "apps" API group, not core ("")

Evidence: Running kubectl api-resources | grep deployments shows:

```
deployments    deploy    apps/v1    true    Deployment
```

Technical Explanation:

- The core API group (represented by empty string "") contains basic Kubernetes resources like Pods, Services, ConfigMaps, and Secrets
- Deployments were moved to the apps API group to better organize application workload resources
- When RBAC checks permissions, it matches both the resource type AND the API group
- A mismatch in either means the permission doesn't apply

Impact: Service account cannot perform ANY operations on deployments because the permission doesn't match the actual resource API group.

Error #2: Namespace Mismatch

Location: RoleBinding metadata

Finding: RoleBinding is in namespace: default but should be in namespace: production

Problem:

- The ServiceAccount is in production namespace
- The Role is in production namespace
- The RoleBinding is in default namespace

Technical Explanation:

- RoleBindings grant permissions within their own namespace
- A RoleBinding in namespace A cannot grant permissions to resources in namespace B
- A RoleBinding in namespace A cannot bind to a ServiceAccount in namespace B
- The RoleBinding must be in the same namespace as the resources it grants access to

Impact: The binding never takes effect because it's looking for the ServiceAccount in the wrong namespace. Kubernetes cannot find the ServiceAccount referenced in the RoleBinding.

Error #3: Cross-Namespace Role Reference

Location: RoleBinding roleRef

Finding: The RoleBinding tries to reference a Role from a different namespace

Problem:

- The RoleBinding is in default namespace
- The Role is in production namespace
- RoleBindings can only reference Roles in the SAME namespace

Technical Explanation:

- Kubernetes RBAC enforces namespace boundaries strictly

- A RoleBinding and its referenced Role must be co-located in the same namespace
- This is a fundamental design principle to maintain namespace isolation
- Cross-namespace references would break the security model

Impact: Kubernetes will not allow this RoleBinding to be created or will ignore it. The API server may reject the configuration.

Step 3: Test Broken Configuration

```
kubectl auth can-i list deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
```

Result: no

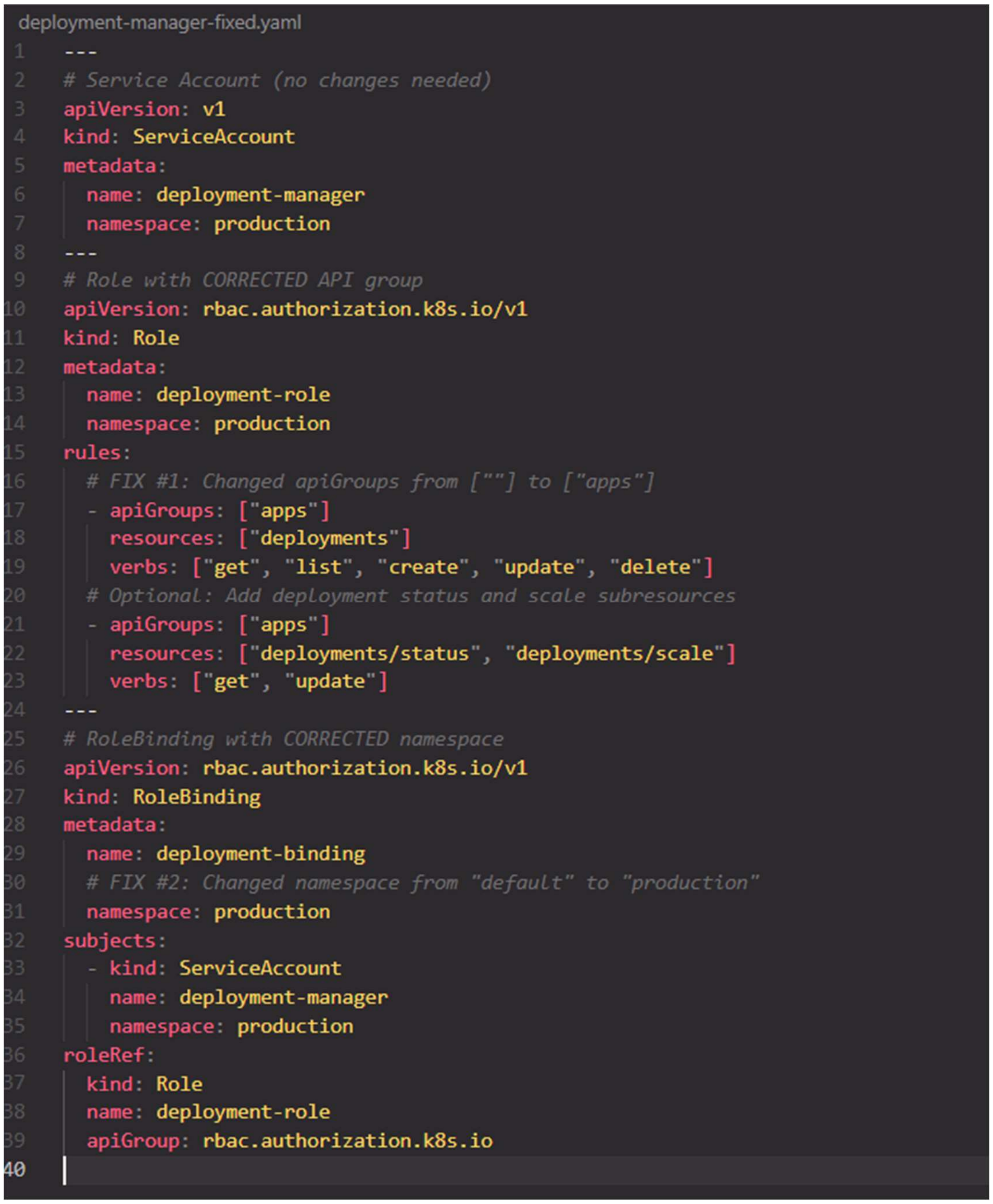
Analysis:

- The command impersonates the service account
- Checks if it has permission to list deployments
- Returns "no" because none of the RBAC rules are effective due to the errors

---

Step 4: Apply Fixes

Corrected YAML Screenshot



deployment-manager-fixed.yaml

```yaml
---
# Service Account (no changes needed)
apiVersion: v1
kind: ServiceAccount
metadata:
  name: deployment-manager
  namespace: production
---
# Role with CORRECTED API group
apiVersion: rbac.authorization.k8s.io/v1
kind: Role
metadata:
  name: deployment-role
  namespace: production
rules:
  # FIX #1: Changed apiGroups from [""] to ["apps"]
  - apiGroups: ["apps"]
    resources: ["deployments"]
    verbs: ["get", "list", "create", "update", "delete"]
  # Optional: Add deployment status and scale subresources
  - apiGroups: ["apps"]
    resources: ["deployments/status", "deployments/scale"]
    verbs: ["get", "update"]
---
# RoleBinding with CORRECTED namespace
apiVersion: rbac.authorization.k8s.io/v1
kind: RoleBinding
metadata:
  name: deployment-binding
  # FIX #2: Changed namespace from "default" to "production"
  namespace: production
subjects:
  - kind: ServiceAccount
    name: deployment-manager
    namespace: production
roleRef:
  kind: Role
  name: deployment-role
  apiGroup: rbac.authorization.k8s.io

```

Changes Made:

1. **Fixed API Group:**

```
# Before
apiGroups: [""]

# After
apiGroups: ["apps"]
```

2. **Fixed Namespace:**

```
# Before
metadata:
  name: deployment-binding
  namespace: default

# After
metadata:
  name: deployment-binding
  namespace: production
```

3. **Ensured Co-location:**
    - ServiceAccount: production namespace ✓
    - Role: production namespace ✓
    - RoleBinding: production namespace ✓

Step 5: Verify Fix

```
kubectl auth can-i list deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
```

Result: yes

Verification Tests:

```
# Test all verbs
kubectl auth can-i get deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
# Result: yes

kubectl auth can-i create deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
# Result: yes

kubectl auth can-i update deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
# Result: yes

kubectl auth can-i delete deployments \
  --as=system:serviceaccount:production:deployment-manager \
  -n production
# Result: yes
```

Impact Analysis

Error	Severity	Impact	Detection Method
Wrong API Group	HIGH	No permissions granted at all	kubectl auth can-i returns no
Namespace Mismatch	HIGH	Binding never takes effect	Describe RoleBinding shows no subjects
Cross-namespace Ref	MEDIUM	Configuration invalid	API server may reject