Kubernetes Cluster Component Security

Controller Manager

Node controller
Job controller
EndpointSlice controller
ServiceAccount controller
Replication Controller : to manage the creation and deletion of pod replicas.
Persistent Volume Controller : Manage the lifecycle of persistent volumes.

How do controllers work:
Monitor : monitors the state of the cluster through the API Server
Analyze : analyze the current state and compare it to the desired state defined in your Kubernetes configurations (like deployment specs or pod definitions)
Reconcile : in case of any fault/discrepancy, take corrective actions to bring the cluster back to the desired state.

Security Policies
Restrict Access : RBAC policies,
Secure Communication : TLS, Regularly rotate keys and certificates.
Audit logs : Enable audit logs, alerts setup in case suspicious activities, regularly monitor.
Controller Health Checks :

Practice Lab:

Step 1: Enable Secure Communication

1. **Generate Certificates**
   - Use a tool like openssl to generate server certificates.

```
       openssl genrsa -out controller-manager.key 2048
       openssl req -new -key controller-manager.key -out
controller-manager.csr -subj "/CN=kube-controller-manager"
       openssl x509 -req -in controller-manager.csr -CA ca.crt -CAkey
ca.key -CAcreateserial -out controller-manager.crt -days 365
```

2. **Configure the Controller Manager to Use Certificates**
   ◦ Edit the Controller Manager manifest (usually located in /etc/kubernetes/manifests/kube-controller-manager.yaml).
   ◦ Add the following flags:

```
      - --tls-cert-file=/etc/kubernetes/pki/controller-manager.crt
      -
--tls-private-key-file=/etc/kubernetes/pki/controller-manager.key
      - --client-ca-file=/etc/kubernetes/pki/ca.crt
```

Step 2: Implement RBAC

1. **Create Roles for Controller Manager**
   ◦ Define roles with specific permissions for the Controller Manager.

```
      apiVersion: rbac.authorization.k8s.io/v1
      kind: ClusterRole
      metadata:
        name: system:kube-controller-manager
      rules:
      - apiGroups: [""]
        resources: ["nodes"]
        verbs: ["get", "list", "watch"]
```

2. **Create RoleBindings**
   ◦ Bind the roles to the Controller Manager.

```
      apiVersion: rbac.authorization.k8s.io/v1
      kind: ClusterRoleBinding
      metadata:
        name: system:kube-controller-manager
      subjects:
      - kind: ServiceAccount
        name: kube-controller-manager
        namespace: kube-system
      roleRef:
        kind: ClusterRole
        name: system:kube-controller-manager
        apiGroup: rbac.authorization.k8s.io
```

Step 3: Enable Audit Logging

1. **Enable Audit Logs**
   - Configure the Controller Manager to log audit events.

```
- --audit-log-path=/var/log/kubernetes/controller-manager-audit.log
      - --audit-policy-file=/etc/kubernetes/audit-policy.yaml
```

2. **Define an Audit Policy**
   - Create a policy file to specify what events to log.

```
      apiVersion: audit.k8s.io/v1
      kind: Policy
      rules:
      - level: Metadata
        resources:
        - group: ""
          resources: ["pods", "nodes"]
```

3. **Apply the Audit Policy**
   - Place the audit policy file in the specified path and restart the Controller Manager.

Above exercise is just for techies, you can try it out and sort out the errors or perform debugging yourself it will not come in exam, as it is Multiple Choise Exam.

By following these steps, you have configured and secured the Kubernetes Controller Manager. You have enabled secure communication, implemented RBAC, and set up audit logging. These practices help protect the Controller Manager from unauthorized access and provide insights into cluster activities for security and compliance.