Kubernetes Cluster Component Security

Scheduler

Control plane component that watches for newly created Pods with no assigned node, and selects a node for them to run on.

Factors taken into account for scheduling decisions include: individual and collective resource requirements, hardware/software/policy constraints, affinity and anti-affinity specifications, data locality, inter-workload interference, and deadlines.

Key Concepts of Scheduling:

Scheduling Process: Evaluates a node that best fits for a pod.
Considering CPU, memory, Node taints and toleration, affinity rules.

Predicates and Priorities: Predicates are checks to filter out nodes that cannot run the pod (e.g., insufficient resources).

Prorities rank the remaining nodes to select most suitable one.

Affinity and Anti-Affinity:

Affinity rules ensure that pods are placed on specific nodes or with other pods.

Anti-affinity rules prevent pods from being placed on certain nodes or with other pods.

Taints and Tolerations:

Taints allow nodes to repel certain pods.

Toleration allows pods to be scheduled on nodes with specific taints.

Resource Requests and Limits:

Pods specify their resource requirements through requests and limits.

The scheduler uses these specifications to ensure efficient resource utilization.

Security Best Practices for Scheduler:

Restrict Access: RBAC Policies, ensure authorizarion.

Secure Communication: TLS, regularly rotate certificates and key.

Audit logs: enable them, setup alerts for suspicios patterns.

Node Isolation: taints and tolerations, critical pods must run on trusted nodes.

Lab Exercise:
Step-by-Step Instructions
Step 1: Enable Secure Communication

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

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

2. **Configure the Scheduler to Use Certificates**
   - Edit the Scheduler manifest (usually located in /etc/kubernetes/manifests/kube-scheduler.yaml).
   - Add the following flags:

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

Step 2: Implement RBAC

1. **Create Roles for Scheduler**
   - Define roles with specific permissions for the Scheduler.

```
      apiVersion: rbac.authorization.k8s.io/v1
      kind: ClusterRole
      metadata:
        name: system:kube-scheduler
      rules:
      - apiGroups: [""]
```

```
  resources: ["nodes", "pods"]
  verbs: ["get", "list", "watch"]
```

2. **Create RoleBindings**
   - Bind the roles to the Scheduler.

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

Step 3: Enable Audit Logging

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

```
- --audit-log-path=/var/log/kubernetes/scheduler-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 Scheduler.

Step 4: Configure Scheduling Policies

1. **Create a Scheduling Policy**
   ◦ Define custom predicates and priorities.

```
apiVersion: kubescheduler.config.k8s.io/v1beta1
kind: KubeSchedulerConfiguration
profiles:
- schedulerName: default-scheduler
  plugins:
    score:
      enabled:
      - name: NodeResourcesBalancedAllocation
      - name: ImageLocality
```

2. **Apply the Scheduling Policy**
   ◦ Edit the Scheduler manifest to include the policy file.

```
- --config=/etc/kubernetes/scheduler-policy.yaml
```

Conclusion

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 Scheduler. You have enabled secure communication, implemented RBAC, set up audit logging, and configured custom scheduling policies. These

practices help protect the Scheduler from unauthorized access and ensure efficient and secure scheduling of workloads.