CKS 2026 Expected Exam – Solutions

Based on expect-exam topics. Many tasks require node/SSH access; solutions are for reference.

---

1. CIS Benchmark – Kubelet (killer-2-7)

Kubelet config (e.g. `/var/lib/kubelet/config.yaml` or via ConfigMap):

- **Authentication:** set anonymous to false (in kubelet config).
- **Authorization:** `mode: Webhook`.

ConfigMap (if used):

```
kubectl -n kube-system edit cm kubelet-config
# Set anonymous: false in authentication section
# Set authorization.mode: Webhook
```

etcd manifest `/etc/kubernetes/manifests/etcd.yaml`:

- **Add:** `--client-cert-auth=true`, `--peer-client-cert-auth=true`
- **Remove:** `--insecure-listen-address`

Then:

```
kubeadm upgrade node phase kubelet-config
service kubelet restart
```

---

2. Remove Anonymous Access to API Server

kube-apiserver manifest:

- `--anonymous-auth=false`
- `--authorization-mode=NODE,RBAC`
- `--enable-admission-plugins=NodeRestriction`

Then remove the ClusterRoleBinding that allows anonymous access (as specified in the question).

---

3. Image Policy Webhook

1. **Config file (JSON):** set `defaultAllow` to `false` (implicit deny).
2. **Kubeconfig:** point to the webhook server URL given in the question.
3. **kube-apiserver manifest:**
4. `--enable-admission-plugins=ImagePolicyWebhook`
5. `--admission-control-config-file=<path of config file>`

---

4. Deployment – readOnlyRootFilesystem

Update the deployment so the container has:

```
securityContext:
  readOnlyRootFilesystem: true
```

---

5. Deployment – Security Context (2 containers)

In each container (or at pod level if required):

```
securityContext:
  runAsUser: 30000
  allowPrivilegeEscalation: false
  readOnlyRootFilesystem: true
```

---

6. Istio Sidecar Injection (killer-2-10)

```
kubectl get ns --show-labels
kubectl label ns team-sedum istio-injection=enabled
kubectl get ns team-sedum --show-labels
kubectl -n team-sedum rollout restart deploy one
kubectl -n team-sedum get pod
kubectl -n team-sedum rollout restart deploy two
kubectl -n team-sedum get pod
```

---

7. NetworkPolicy – Deny All (development)

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: deny-all-traffic
  namespace: development
spec:
  podSelector: {}
  policyTypes:
  - Ingress
  # Add - Egress if question asks to deny egress too
```

---

8. NetworkPolicy – Allow from Stage and QA (naboo)

```
apiVersion: networking.k8s.io/v1
kind: NetworkPolicy
metadata:
  name: allow-from-stage-and-qa
  namespace: naboo
spec:
  podSelector: {}
```

```
  policyTypes:
  - Ingress
  ingress:
  - from:
    - namespaceSelector:
        matchLabels:
          name: qa
    - podSelector:
        matchLabels:
          environmental: stage
```

---

9. Upgrade Kubernetes (killer-2-17)

```
kubectl get node
sudo -i
kubelet --version
kubeadm version
kubectl drain <node-name> --ignore-daemonsets
ssh <node-name>
apt update
apt-mark hold kubeadm
kubeadm upgrade node
apt-mark unhold kubectl kubelet
apt install kubelet=<version> kubectl=<version>
service kubelet restart
service kubelet status
apt-mark hold kubelet kubectl
exit
kubectl get node
kubectl uncordon <node-name>
kubectl get node
```

---

10. ServiceAccount Token Expiration (killer-1-4)

Example deployment changes:

- Annotation: `token-lifetime: "1200"`
- `serviceAccountName: stream-multiplex`
- `automountServiceAccountToken: false`
- Mount ServiceAccount token at `/var/run/secrets/custom/` with `expirationSeconds: 1200`

```
# stream-multiplex deployment (team-coral)
spec:
  template:
    metadata:
      annotations:
        token-lifetime: "1200"
    spec:
      serviceAccountName: stream-multiplex
      automountServiceAccountToken: false
      containers:
        - name: httpd
          volumeMounts:
            - name: token-volume
              mountPath: /var/run/secrets/custom
```

```
              readOnly: true
      volumes:
        - name: token-volume
          projected:
            sources:
              - serviceAccountToken:
                  path: token
                  expirationSeconds: 1200
```

---

11. TLS Secret – Create and Mount

```
kubectl create secret tls code-secret \
  --cert=/root/custom-cert.crt \
  --key=/root/custom-key.key \
  -n code
kubectl edit deployment code-server -n code
```

In deployment:

- **spec.template.spec:** add volume from secret:

```
volumes:
  - name: secret-volume
    secret:
      secretName: code-secret
```

- **spec.template.spec.containers:** add volumeMount:

```
volumeMounts:
  - name: secret-volume
    mountPath: /etc/code/tls
    readOnly: true
```

---

12. Docker / Runtime Security (moc-3-1)

Remove user from docker group:

```
sudo gpasswd -d <user> docker
newgrp $(id -gn)
# or: groups <user> / id <user>
```

Secure Docker:

- `sudo chown root:root /var/run/docker.sock`
- In Docker service: `ExecStart=/usr/bin/dockerd --group=root`
- `/etc/docker/daemon.json` : only `"hosts": ["unix:///var/run/docker.sock"]` (no TCP)
- `sudo systemctl daemon-reexec` , `daemon-reload` , `restart docker`

---

13. Falco – Custom Rule and Scale to Zero

- Find the pod using the path (e.g. under `/dev/x`).
- Add Falco rule (condition matching that path, description/output as required).
- Scale the deployment that uses that path to zero: `kubectl scale deployment <name> --replicas=0`.

---

14. ImagePolicyWebhook – AdmissionConfiguration

- Create AdmissionConfiguration with ImagePolicyWebhook (kubeConfigFile, allowTTL, denyTTL, retryBackoff, defaultAllow).
- Fix paths in admission config and kubeconfig (server URL, cert paths).
- Mount config dir in API server and set:
- `--enable-admission-plugins=ImagePolicyWebhook`
- `--admission-control-config-file=/path/to/admission_configuration.yaml`

Example admission config:

```
apiVersion: apiserver.config.k8s.io/v1
kind: AdmissionConfiguration
plugins:
- name: ImagePolicyWebhook
  configuration:
    imagePolicy:
      kubeConfigFile: /etc/kubernetes/pki/admission_kube_config.yaml
      defaultAllow: false
```

---

15. Pod Security Standard – Restricted (mok-1-10)

In the deployment (e.g. `web-server` in `restricted`):

- `allowPrivilegeEscalation: false`
- `capabilities.drop: ["ALL"]`
- `runAsNonRoot: true`
- `seccompProfile.type: RuntimeDefault`
- Remove `runAsUser: 0`

```
kubectl edit deployment web-server -n restricted
```

---

16. Audit Log Policy (killer-1-17 / moc-1-15)

- Configure audit to keep only one backup of logs (log age / backup count as required).
- Policy: Secret resources → Metadata; `system:nodes` userGroups → RequestResponse; all other → Metadata.
- Add volume and volumeMount for policy file and log file in API server pod.
- After changes: `echo > /etc/kubernetes/audit/logs/audit.log` (or equivalent) so the log only contains entries per the new policy.

Extra Topics from expect-exam

Dockerfile / Deployment best practices (killer-2-3): Deployment: set `allowPrivilegeEscalation: false`,

`capabilities.drop: ["ALL"]`. Dockerfile: use tagged base image and non-root user (e.g. `USER nobody` at the end, after package installs as root).

Trivy / image scan: Find deployment with the given image, run Trivy scan, generate SBOM (e.g. `trivy image --format json` or similar). Remove pods/deployments with Critical/High vulnerabilities; backup manifests first.

Ingress (killer-1-15): Example for `rocket-server` in namespace `space` with TLS and nginx Ingress class – see expect-exam.txt for full YAML (host, TLS secret, backend service, path).