

Question 1: Falco Runtime Security Detection

Domain: Monitoring, Logging & Runtime Security (20%)

Difficulty: Medium

Task

Three deployments exist in namespace `apps`: `nvidia-gpu`, `cpu`, and `ollama`. Pods from one of these deployments are accessing `/dev/mem`, causing memory issues on the node.

1. Use Falco to identify which pod is accessing `/dev/mem`
2. Scale the related deployment to 0 replicas

Question 2: Worker Node Kubernetes Upgrade

Domain: Cluster Hardening (15%)

Difficulty: Medium

Task

A worker node named `node01` is running kubelet version 1.34.0. The control plane is already at version 1.34.1.

Upgrade the worker node `node01` to version 1.34.1

Question 3: Ingress with TLS and HTTP to HTTPS Redirect

Domain: Cluster Setup (15%)

Difficulty: Easy

Task

A TLS secret named `tls-secret` already exists in the namespace `secure-app`.

A service named `secure-service` is already running on port 80.

1. Create an Ingress resource in the `secure-app` namespace
2. Configure the Ingress to use the existing TLS secret `tls-secret`
3. Configure the Ingress to redirect all HTTP traffic to HTTPS
4. The Ingress should route traffic for host `secure.example.com` to the `secure-service`

Question 4: SBOM with SPDX Format

Domain: Supply Chain Security (20%)

Difficulty: Easy

Task

Generate a Software Bill of Materials (SBOM) for a specified container image.

1. Use the `bom` tool to generate an SBOM in SPDX format
2. Analyze the container image provided in the question
3. Save the output to `/opt/course/<N>/sbom.spdx`

Question 5: Create TLS Secret

Domain: Cluster Setup (15%)

Difficulty: Very Easy

Task

Create a TLS secret using the provided certificate and key files.

- Certificate path: `/opt/course/<N>/tls.crt`
- Key path: `/opt/course/<N>/tls.key`

- Secret name: `my-tls-secret`
- Namespace: `secure-ns`

Question 6: Docker Daemon Security Hardening

Domain: System Hardening (15%)

Difficulty: Medium

Task

SSH to the specified cluster node and perform the following tasks to secure the Docker daemon:

1. Remove user `developer` from the `docker` group (do not remove from any other group)
2. Reconfigure the Docker daemon to ensure the socket file `/var/run/docker.sock` is owned by group `root`
3. Reconfigure the Docker daemon to ensure it does not listen on any TCP port
4. Restart the Docker daemon
5. Ensure the Kubernetes cluster is healthy after your changes

Question 7: Network Policy - Deny Ingress and Allow from Specific Namespace with Pod Labels

Domain: Cluster Setup (10%)

Difficulty: Medium-Hard

Task

You have two namespaces: `prod` and `data`.

- The namespace `data` is labeled with `env: data`
1. Create a NetworkPolicy named `deny-all-ingress` in the `prod` namespace that denies ALL ingress traffic to all pods
 2. Create a NetworkPolicy named `allow-from-prod` in the `data` namespace that allows ingress traffic ONLY from:
 - Pods that are in the `prod` namespace
 - AND those pods must have the label `env: prod`

Question 8: ServiceAccount Token Mounting with Projected Volume

Domain: Cluster Hardening (15%)

Difficulty: Medium

Task

A ServiceAccount named `backend-sa` exists in namespace `secure`.

A Deployment named `backend-deploy` exists in the same namespace using this ServiceAccount.

1. Edit the ServiceAccount `backend-sa` to disable automatic token mounting (`automountServiceAccountToken: false`)
 2. Edit the Deployment `backend-deploy` to manually mount the ServiceAccount token using a projected volume:
 - Create a projected volume named `token`
 - Mount the token at path `/var/run/secrets/kubernetes.io/serviceaccount` (or the path specified in the exam)
 - The volume mount must be read-only
-

Question 9: Configure Kubernetes Auditing

Domain: Monitoring, Logging and Runtime Security (20%)

Difficulty: Medium-Hard

Task

An audit policy file is provided at a specific path (e.g., `/etc/kubernetes/audit/audit-policy.yaml`).

1. Configure the kube-apiserver to enable auditing:

- Set the audit policy file path using `--audit-policy-file`
- Set the audit log path using `--audit-log-path` (e.g., `/var/log/kubernetes/audit/audit.log`)
- Set `--audit-log-maxage` to `2`
- Set `--audit-log-maxbackup` to `10`

2. Edit the audit policy file to add rules for specific resources:

- Add a rule to log `secrets` at `Metadata` level
- Add a rule to log `configmaps` at `Metadata` level
- Add a rule to log `namespaces` at `RequestResponse` level (or as specified)
- Rules should specify `group: ""` (core API group) and the resource names in the `resources` array

3. Ensure the API server mounts the necessary volumes for the audit policy file and log directory

Question 10: ImagePolicyWebhook Admission Controller Setup

Domain: Supply Chain Security (20%)

Difficulty: Medium-Hard

Task

Configuration files for ImagePolicyWebhook are provided at a specific path (e.g., `/etc/kubernetes/epconfig/`).

1. Enable the `ImagePolicyWebhook` admission controller in the kube-apiserver:

- Add `ImagePolicyWebhook` to `--enable-admission-plugins`
- Set `--admission-control-config-file` to point to the admission configuration file

2. Edit the ImagePolicyWebhook configuration file to configure:

- Set `defaultAllow` to `false` (deny images by default when webhook is unreachable)
- Configure the `kubeConfigFile` path pointing to the kubeconfig for the backend

3. Edit the kubeconfig file for the webhook backend:

- Set the `server` URL to the external image policy service (e.g., `https://image-policy-webhook.default.svc:443/image_policy`)
- Configure the certificate authority and client certificates if required

4. Ensure the API server mounts the necessary volumes for the configuration files

Question 11: Pod Security Admission (PSA) - Identify and Delete Non-Compliant Pods

Domain: Minimize Microservice Vulnerabilities (20%)

Difficulty: Medium

Task

A namespace (e.g., `team-blue`) has Pod Security Admission configured with the `restricted` level in `enforce` mode. Several pods exist in this namespace, but some were created before the policy was enforced and do not comply with the `restricted` Pod Security Standard.

1. Use `kubectl label --dry-run=server` to identify pods that violate the Pod Security Standard:
 - `kubectl label --dry-run=server --overwrite ns <namespace> pod-security.kubernetes.io/enforce=restricted`
 2. Review the warnings to identify non-compliant pods
 3. Delete the pods that do not comply with the `restricted` policy
 4. The compliant pods should remain running
-

Question 12: Dockerfile and Deployment Security Best Practices (Static Manual Analysis)

Domain: Supply Chain Security (20%)

Difficulty: Medium

Task

A Dockerfile and a Kubernetes Deployment manifest are provided at a specific path (e.g., `/opt/course/<N>/`). Both files contain security issues that do not follow best practices.

1. Analyze the Dockerfile and fix the security issues:
 - Running as root user (add `USER <username>` instruction with a non-root user)
 - Using `latest` tag for base image (change to specific version like `nginx:1.25.3-alpine`)
 - Using `ADD` instead of `COPY` for local files (use `COPY` when not extracting archives)
 2. Analyze the Deployment manifest and fix the securityContext issues:
 - Container running as privileged (set `privileged: false`)
 - Container can escalate privileges (set `allowPrivilegeEscalation: false`)
 - Container not running as non-root (set `runAsNonRoot: true`)
 - Root filesystem is writable (set `readOnlyRootFilesystem: true` for immutability)
-

Question 13: Kubelet Security Configuration

Domain: Cluster Hardening (15%)

Difficulty: Medium

Task

SSH to the specified node and secure the kubelet configuration file located at `/var/lib/kubelet/config.yaml`.

1. Disable anonymous authentication:
 - Set `authentication.anonymous.enabled` to `false`
2. Enable webhook authentication (for token-based auth):
 - Set `authentication.webhook.enabled` to `true`

3. Enable webhook authorization:

- Set `authorization.mode` to `Webhook`

4. Restart the kubelet service:

- `systemctl restart kubelet`

5. Verify the node is in Ready state after changes

Question 14: Ensure Immutability of Containers at Runtime

Domain: Monitoring, Logging and Runtime Security (20%)

Difficulty: Medium

Task

A Deployment (e.g., `nginx`) exists in a specified namespace. Modify the Deployment to ensure the container filesystem is immutable at runtime.

1. Edit the Deployment to add `readOnlyRootFilesystem: true` in the container's `securityContext`
2. The pod may fail to start because certain applications need writable directories. Check the logs to identify the required paths (e.g., `/var/cache/nginx`, `/var/run`, `/tmp`)
3. Add `emptyDir` volumes and mount them to the writable paths required by the application:
 - For `nginx`: mount `emptyDir` to `/var/cache/nginx` and `/var/run`
4. Ensure the pod is running successfully after the changes