DUMPSARENA

Red Hat Certified Specialist in OpenShift Automation and Integration

RedHat EX380

Version Demo

Total Demo Questions: 4

Total Premium Questions: 42

Buy Premium PDF

https://dumpsarena.co

sales@dumpsarena.co

sales@dumpsarena.co
dumpsarena.co

Topic Break Down

Topic	No. of Questions
Topic 1, Configure and manage OpenShift Container Platform	32
Topic 2, Deploy and manage applications on OpenShift Container Platform	2
Topic 3, Configure and manage OpenShift GitOps	6
Topic 4, Configure and manage OpenShift event-driven architectures	2
Total	42

QUESTION NO: 1 - (SIMULATION)

Logging Configuration – Configure ClusterLogging in Web Console

ANSWER: See the explanation for the answer

Explanation:

In the OpenShift web console, update the existing `ClusterLogging` custom resource so that its collector type is `vector`.

1. Open **Operators → Installed Operators**.
2. Select the **Red Hat OpenShift Logging** operator (typically in the `openshift-logging` project).
3. Open the **ClusterLogging** tab and select the existing ClusterLogging instance.
4. Choose **Actions → Edit ClusterLogging** (or open the **YAML** tab/editor).
5. Set the collection configuration to:

```
spec:
  collection:
    type: vector
```

If `spec.collection` already exists, change only its `type` value to `vector`. Save the resource and allow the logging operator to reconcile the updated collector configuration.

QUESTION NO: 2 - (SIMULATION)

Configure and synchronize OpenShift groups from LDAP (group sync)

Task Information : Create an LDAP group-sync config, run a one-time sync, and confirm groups exist in OpenShift.

ANSWER: See the explanation for the answer

Explanation:

Create an LDAP sync configuration on the host where the `oc` command will be run. For an RFC2307-style LDAP directory, create `groupsync.yaml` with the LDAP server values appropriate for the environment:

```
apiVersion: v1
kind: LDAPSyncConfig
url: ldap://ldap.example.com:389
bindDN: "cn=openshift-sync,ou=serviceaccounts,dc=example,dc=com"
bindPassword:
  file: /path/to/ldap-bind-password
insecure: false
ca: /path/to/ldap-ca.crt
rfc2307:
  groupsQuery:
    baseDN: "ou=groups,dc=example,dc=com"
    scope: sub
    derefAliases: never
    timeout: 0
    filter: "(objectClass=groupOfNames)"
    pageSize: 0
  groupUIDAttribute: dn
  groupNameAttributes:
  - cn
  groupMembershipAttributes:
  - member
  usersQuery:
    baseDN: "ou=users,dc=example,dc=com"
    scope: sub
    derefAliases: never
    timeout: 0
```

```
    filter: "(objectClass=person)"
    pageSize: 0
  userUIDAttribute: dn
  userNameAttributes:
  - uid
  tolerateMemberNotFoundErrors: false
  tolerateMemberOutOfScopeErrors: false
```

Store the LDAP bind password in the referenced local file, for example:

```
printf '%s' 'LDAP_BIND_PASSWORD' > /path/to/ldap-bind-password
chmod 600 /path/to/ldap-bind-password
```

First preview the result (do not use `--confirm`):

```
oc adm groups sync --sync-config=groupsync.yaml
```

When the preview is correct, perform the one-time synchronization:

```
oc adm groups sync --sync-config=groupsync.yaml --confirm
```

Confirm that OpenShift group objects and their LDAP-derived members exist:

```
oc get groups
oc describe group <group-name>
```

The actual LDAP URL, bind DN, CA path, base DNs, object classes, and member attributes must match the LDAP server. If the directory uses a different group schema (for example Active Directory), use the matching `activeDirectory` or `augmentedActiveDirectory` section instead of `rfc2307`.

QUESTION NO: 3 - (SIMULATION)

Deploy Event Router and capture Kubernetes events in logging

Task Information : Deploy an event router so Kubernetes events are recorded as logs, then trigger events and confirm they appear in logging queries.

ANSWER: See the explanation for the answer

Explanation:

Create an Event Router in `openshift-logging`. Its service account must be able to list/watch events cluster-wide, and the router must write to standard output so that the OpenShift logging collector forwards it.

```
cat > eventrouter.yaml <<'EOF'
apiVersion: v1
kind: ServiceAccount
metadata:
  name: eventrouter
  namespace: openshift-logging
---
apiVersion: rbac.authorization.k8s.io/v1
kind: ClusterRole
metadata:
  name: eventrouter
rules:
- apiGroups: [""]
  resources: ["events"]
  verbs: ["get", "list", "watch"]
- apiGroups: ["events.k8s.io"]
  resources: ["events"]
  verbs: ["get", "list", "watch"]
---
apiVersion: rbac.authorization.k8s.io/v1
```

```
kind: ClusterRoleBinding
metadata:
  name: eventrouter
roleRef:
  apiGroup: rbac.authorization.k8s.io
  kind: ClusterRole
  name: eventrouter
subjects:
- kind: ServiceAccount
  name: eventrouter
  namespace: openshift-logging
---
apiVersion: v1
kind: ConfigMap
metadata:
  name: eventrouter
  namespace: openshift-logging
data:
  config.json: |
    {"sink":"glog"}
---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: eventrouter
  namespace: openshift-logging
spec:
  replicas: 1
  selector:
    matchLabels:
      app: eventrouter
  template:
    metadata:
      labels:
        app: eventrouter
    spec:
      serviceAccountName: eventrouter
      containers:
      - name: eventrouter
        image: registry.redhat.io/openshift-logging/eventrouter-rhel8:v0.4
        args: ["-v=3", "-logtostderr"]
        volumeMounts:
        - name: config
          mountPath: /etc/eventrouter
          readOnly: true
      volumes:
      - name: config
        configMap:
          name: eventrouter
EOF
oc apply -f eventrouter.yaml
oc -n openshift-logging rollout status deployment/eventrouter
```

Confirm that the router is running and can emit event records:

```
oc -n openshift-logging get pods -l app=eventrouter
oc -n openshift-logging logs deployment/eventrouter --tail=20
```

Generate a few Kubernetes events. For example:

```
oc -n default run evtest --image=registry.access.redhat.com/ubi9/ubi-minimal \
  --restart=Never -- /bin/sh -c 'exit 0'
oc -n default wait --for=condition=Ready pod/evtest --timeout=30s || true
oc -n default delete pod evtest --ignore-not-found
```

In the OpenShift console, open **Observe** → **Logs**, select the logging application log source/store, and query for the router output, for example `kubernetes.namespace_name = "openshift-logging" AND kubernetes.pod_name =~ "eventrouter"`. Search the returned JSON log messages for `evtest`, `default`, or event reasons such as `Scheduled`, `Pulling`, `Created`, and `Killing`.

The important validation is that `oc logs` shows event JSON from the Event Router and the same records become searchable in the logging query UI. The exact Event Router image tag can differ by installed OpenShift Logging version; use the supported `eventrouter-rhel8` tag mirrored/available in the cluster if `v0.4` is not available.

QUESTION NO: 4 - (SIMULATION)

Service Accounts and RBAC – Grant Cluster Reader Role

ANSWER: See the explanation for the answer

Explanation:

Grant the `cluster-reader` ClusterRole to the `audit` service account in the `auth-audit` namespace:

```
oc adm policy add-cluster-role-to-user cluster-reader system:serviceaccount:auth-audit:audit
```

This creates a cluster-wide RBAC binding for the service account identity `system:serviceaccount:auth-audit:audit`, allowing read-only access as defined by the `cluster-reader` role.

Optionally verify the assigned role:

```
oc adm policy who-can get pods --all-namespaces
oc get clusterrolebinding -o wide | grep audit
```