Red Hat

Red Hat build of Keycloak 26.0 on OpenShift Create

Deployment Document: Keycloak with PostgreSQL on OpenShift

Prepared by: *Sunny Rampalli*
Date: *May 13, 2025*
Cluster Environment: *OpenShift v4.18*
Namespace: `keycloak-sso`

---

1. Objective

Deploy and configure Keycloak (Red Hat build of Keycloak) with a PostgreSQL database backend in an OpenShift environment using secure TLS communication and Kubernetes secrets for credentials.

2. TLS & Secret Configuration

2.1 TLS Secret Creation

```
Unset
oc create secret tls my-tls-secret \
  --cert=apps.crt \
  --key=apps.key \
  -n keycloak-sso
```

2.2 Database Credentials Secret

```
Unset
oc create secret generic keycloak-db-secret \
  --from-literal=username=keycloak \
  --from-literal=password=keycloak \
  -n keycloak-sso
```

3. PostgreSQL Deployment

Resource File: **`postgres.yaml`**

StatefulSet + PVC + Service

```
Unset
apiVersion: apps/v1
kind: StatefulSet
metadata:
  name: postgresql-db
spec:
  serviceName: postgresql-db-service
  replicas: 1
```

```yaml
  selector:
    matchLabels:
      app: postgresql-db
  template:
    metadata:
      labels:
        app: postgresql-db
    spec:
      containers:
        - name: postgresql-db
          image: postgres:latest
          ports:
            - containerPort: 5432
              name: postgres
          env:
            - name: POSTGRES_USER
              valueFrom:
                secretKeyRef:
                  name: keycloak-db-secret
                  key: username
            - name: POSTGRES_PASSWORD
              valueFrom:
                secretKeyRef:
                  name: keycloak-db-secret
                  key: password
            - name: POSTGRES_DB
              value: keycloak
            - name: PGDATA
              value: /data/pgdata
          volumeMounts:
            - name: postgres-storage
              mountPath: /data
  volumeClaimTemplates:
    - metadata:
        name: postgres-storage
      spec:
        accessModes: [ "ReadWriteOnce" ]
        resources:
          requests:
            storage: 5Gi
---
apiVersion: v1
kind: Service
metadata:
  name: postgresql-db-service
spec:
  selector:
    app: postgresql-db
  type: ClusterIP
  ports:
    - port: 5432
      targetPort: 5432
```

4. Keycloak Deployment

Resource File: `rhbk.yaml`

```
Unset
apiVersion: k8s.keycloak.org/v2alpha1
kind: Keycloak
metadata:
  name: sso-keycloak
  namespace: keycloak-sso
  labels:
    app: sso
spec:
  http:
    tlsSecret: my-tls-secret
  hostname:
    hostname: keycloak.apps.ocp4.ipa.prodevans.com
  db:
    vendor: postgres
    host: postgresql-db-service
    usernameSecret:
      name: keycloak-db-secret
      key: username
    passwordSecret:
      name: keycloak-db-secret
      key: password
  instances: 1
```

5. Routes and Services

Keycloak Services

```
Unset
oc get svc
```

Name	Type	Port(s)	Cluster IP
postgresql-db-service	ClusterIP	5432/TCP	172.30.168.66
sso-keycloak-service	ClusterIP	8443/TCP, 9000/TCP	172.30.142.229
sso-keycloak-discovery	ClusterIP	7800/TCP	None

Ingress Route

```
Unset
oc get route
```

Name	Host	Port	Termination
sso-keycloak-ingress-jqghx keycloak.apps.ocp4.ipa.prodevans.com	https	passthrough/Redirect	

6. Access and Admin Role Setup

Admin CLI Access using `kcadm.sh`

```
Unset
./kcadm.sh add-roles --uusername admin --rolename admin -r master \
  --config /tmp/kcadm.config --insecure
```

Note: TLS verification is skipped in non-production. For production, configure a truststore properly.

Verify Roles

```
Unset
./kcadm.sh get-roles --uusername admin -r master \
  --config /tmp/kcadm.config --insecure
```

Example Output:

```
Unset
[
  {
    "name": "admin",
    "composite": true
  },
  {
    "name": "default-roles-master",
    "composite": true
  }
]
```

---

7. Resource Status Summary

```
Unset
oc get all -n keycloak-sso
```

Resource	Name	Status
Pod	postgresql-db-0	Running

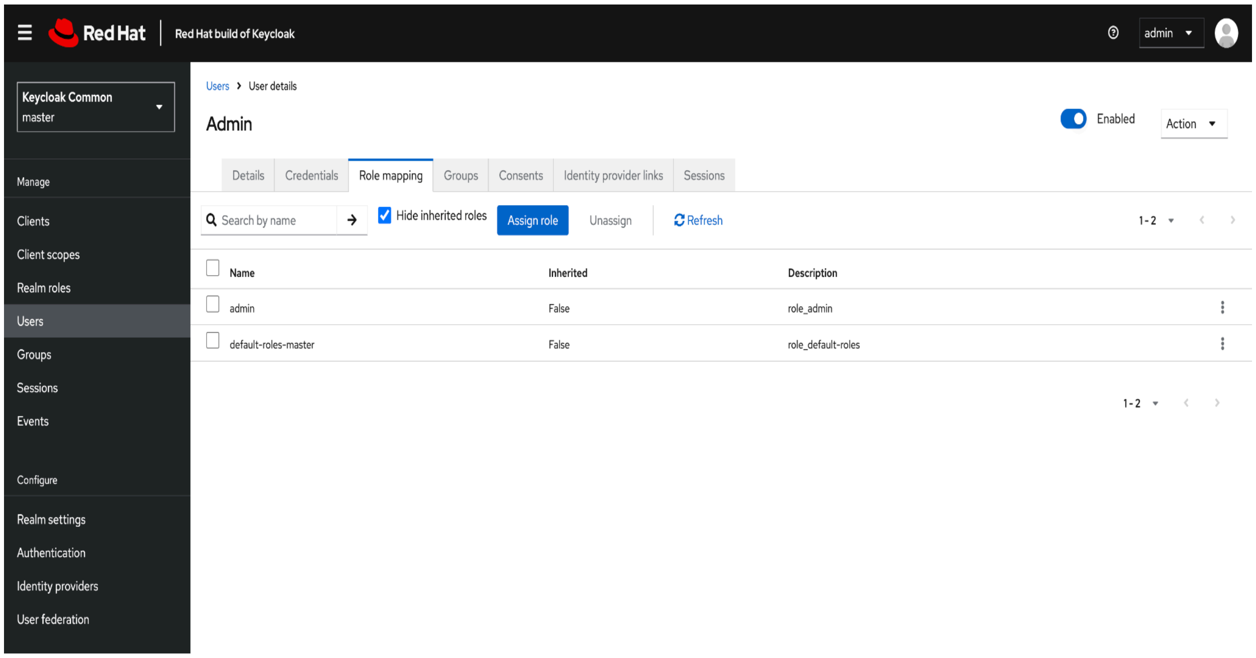
Pod	sso-keycloak-0	Running
StatefulSet	postgresql-db	1/1
StatefulSet	sso-keycloak	1/1
Deployment	rhbk-operator	1/1
PersistentVolumeClaim	postgres-storage-*	Bound

8. Notes

- PVC is backed by NFS (RWX mode).
- TLS is configured via passthrough route; browser warning may occur without proper certs.
- All passwords and usernames are stored securely via Kubernetes secrets.
- PostgreSQL is deployed as a `StatefulSet` for persistence.

9. Recommendations for Production

- Replace `latest` tag for PostgreSQL image with a fixed, tested version.
- Use a proper CA-signed TLS certificate.
- Enable truststore in `kcadm.sh` usage.
- Configure Keycloak backup and monitoring.
- Implement RBAC and audit logging.



Admin

Name	Inherited	Description
admin	False	role_admin
default-roles-master	False	role_default-roles