Rapport: Utmaningar och Lösningar i CI/CD-projektet

Sammanfattning

- Under utvecklingen av en CI/CD-lösning för att bygga och deploya en Nginx-applikation till Kubernetes med GitHub Actions, GHCR och Ansible, uppstod flera tekniska utmaningar.
- Denna rapport sammanfattar problemen, deras orsaker och de lösningar som implementerades.

1. Autentisering mot GHCR misslyckades

Orsak:

- Felmeddelande:: "Username and password required" samt "denied: denied" vid inloggning.
- Felaktig eller saknad autentisering vid användning av GitHub Actions.

Lösning:

- Skapade en Personal Access Token (PAT) med rätt behörigheter (write:packages, read:packages, repo).
- Lagrade PAT som en GitHub Secret (GHCR_PAT).
- Uppdaterade GitHub Actions-workflow att använda PAT istället för GITHUB_TOKEN.

2. ImagePullBackOff vid Kubernetes-deployment

Orsak:

- Kubernetes kunde inte hämta containerbilden från GHCR på grund av saknad autentisering.
- imagePullSecrets saknades eller försvann efter en omstart av Minikube.

Lösning:

- Skapade en Kubernetes-hemlighet (ghcr-secret) för autentisering.
- Säkerställde att imagePullSecrets fanns i deployment-konfigurationen.
- Om hemligheten försvann efter Minikube-restart, återskapades den manuellt.

3. Felaktigt repository-namn vid Docker build

Orsak:

- GHCR kräver att repository-namn är i gemener, men GitHub Actions github.repository kan innehålla versaler.

- Felmeddelande: "repository name must be lowercase".

Lösning:

- Konverterade repository-namnet till små bokstäver i GitHub Actions-workflow innan build och push.

4. Kubernetes använde gamla pods efter uppdateringar

Orsak:

- Kubernetes cachelagrade bilden och drog inte alltid den senaste versionen.

Lösning:

- Använde imagePullPolicy: Always i deployment-konfigurationen.
- Tvingade omstart av pods efter uppdateringar.
- Rensade cache genom att starta om Minikube vid behov.

5. Ansible Playbook kunde inte hämta senaste bildtaggen från GHCR

Orsak:

- API-anropet misslyckades med "authentication required" eller "invalid token".
- PAT-tokenen saknade rättigheter för att hämta tags från GHCR.

Lösning:

- Genererade en ny GitHub Personal Access Token (PAT) med rätt behörigheter (read:packages, write:packages).
- Uppdaterade Ansible-playbooken för att hämta en giltig access-token innan API-anropet.

- Lade till en Ansible-task för att hämta den senaste bildtaggen från GHCR:

```
- name: Fetch the latest image tag from GHCR
  shell: >
        TOKEN=$(curl -s -u "{{ lookup('env', 'GITHUB_USERNAME') }}:{{ lookup('env',
'PERSONAL_ACCESS_TOKEN') }}"

"https://ghcr.io/token?scope=repository:joyjohansson/ci-cd-docker-kubernetes-ansible/my-nginx:pull"
| jq -r .token) &&
        curl -s -H "Authorization: Bearer $TOKEN" \
        "https://ghcr.io/v2/joyjohansson/ci-cd-docker-kubernetes-ansible/my-nginx/tags/list" |
        jq -r '.tags | map(select(test("^v[0-9]+\\.[0-9]+\\.[0-9]+$"))) | sort | last'
  register: latest_ta
```

6. Ansible Playbook misslyckades med anslutning till localhost**

Orsak:

- Ansible försökte ansluta via SSH, trots att deployment kördes lokalt.

Lösning:

- Ändrade Ansible-playbooken för att använda lokal anslutning (connection: local).
- Verifierade syntaxen innan körning med ansible-playbook --syntax-check.

Reflektion:

Jag är medveten om att koderna och skripten i detta projekt inte är optimala och att det finns flera förbättringsåtgärder att implementera. Mina nuvarande kunskaper är begränsade, och jag har använt resurser såsom Stack Overflow, ChatGPT och Verifa som stöd under arbetet. Fokus för denna uppgift har dock inte varit att skriva perfekt kod, utan snarare att förstå hur verktygen integreras och används i en pipeline. Efter diskussion med min lärare har jag dragit slutsatsen att ju fler koder och skript jag läser och analyserar, desto lättare kommer det att bli att i framtiden skriva kod enligt "best practices". Erfarenhet och kontinuerligt lärande kommer att spela en avgörande roll i att utveckla mer robusta och effektiva lösningar framöver.

Slutsats:

Efter att ha övervunnit dessa utmaningar fungerar nu CI/CD-pipelinen enligt följande:

1. Pipeline bygger, taggar och pushar Docker-bilder till GHCR.
2. Ansible hanterar rolling updates och rollback vid fel.
3. Minikube kan köra tjänsten via minikube service nginx-service