Kubernetes Gateway API 測試

您說得對，為了確保這份文件是「自包含 (Self-contained)」且可立即執行的，必須包含後端應用的部署宣告。

身為架構師，在測試金絲雀發佈時，我們會使用具備 不同環境變數 的 Pod 來區分版本。以下是完整的 `apps.yaml` 內容與更新後的部署流程。

---

1. 應用程式定義層：`apps.yaml`

此檔案定義了兩個版本的 Deployment 與對應的 Service。我們使用 `nginxdemos/hello` 映像檔，因為它會直接在網頁內容中顯示 Server Name，方便我們觀察流量去向。

YAML

```yaml
# --- Version 1 (90% 流量) ---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app-v1
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
      version: v1
  template:
    metadata:
      labels:
        app: my-app
        version: v1
    spec:
      containers:
      - name: nginx
        image: nginxdemos/hello:plain-text
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app-v1
spec:
  selector:
    app: my-app
    version: v1
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
---
# --- Version 2 (10% 權重測試) ---
apiVersion: apps/v1
kind: Deployment
metadata:
  name: my-app-v2
spec:
  replicas: 2
  selector:
    matchLabels:
      app: my-app
      version: v2
  template:
    metadata:
```

```
      labels:
        app: my-app
        version: v2
    spec:
      containers:
      - name: nginx
        image: nginxdemos/hello:plain-text
        ports:
        - containerPort: 80
---
apiVersion: v1
kind: Service
metadata:
  name: my-app-v2
spec:
  selector:
    app: my-app
    version: v2
  ports:
  - protocol: TCP
    port: 80
    targetPort: 80
```

---

2. 完整部署與測試標準 SOP (全流程)

第一階段：基礎資源與控制器

Bash

```
# 1. 安裝 Gateway API 與 Traefik CRDs
kubectl apply -f https://github.com/kubernetes-sigs/gateway-api/releases/download/v1.0.0/standard-install.yaml
kubectl apply -f https://raw.githubusercontent.com/traefik/traefik/v3.0/docs/content/reference/dynamic-configuration/kubernetes-crd-definition-v1.yml

# 2. 部署 Traefik (對齊 8000 埠)
helm upgrade --install traefik traefik/traefik \
  --namespace traefik --create-namespace \
  --set ports.web.nodePort=30080 \
  --set ports.web.port=8000 \
  --set service.type=NodePort \
  --set providers.kubernetesGateway.enabled=true \
  --set additionalArguments="{--providers.kubernetesgateway,--entrypoints.web.address=:8000}" \
  --skip-crds
```

第二階段：應用程式與路由套用

Bash

```
# 3. 部署後端 Pods 與 Services
kubectl apply -f apps.yaml

# 4. 部署 Gateway API 路由 (確保 Port 設為 8000)
kubectl apply -f gateway-api.yml
```

---

3. 最終驗證與自動化分析

身為系統建設負責人，我們會使用以下腳本進行最後的數據收集，驗證權重分配是否精確。

A. 檢查 Endpoints 狀態

確保每個版本都有對接到 Pod IP：

Bash

```
kubectl get endpoints my-app-v1 my-app-v2
```

B. 執行金絲雀流量分析

Bash

```
# 執行 20 次測試並計算結果
for i in {1..20}; do
  curl -s -H "Host: localhost" http://localhost | grep "Server name"
done
```

---

4. 維運經驗總結 (Lead's Summary)

這套全內容方案解決了您在實驗中遇到的所有核心技術挑戰：

1. **埠號對齊**：解決了 `Empty reply` 與 `PortUnavailable` 問題。
2. **所有權衝突**：透過重置 Gateway 與正確使用 Helm 參數規避了 `SSA Conflict`。
3. **命名空間授權**：透過 `from: All` 解決了跨 Namespace 404 的問題。
4. **內容識別**：修正了 `grep` 關鍵字，確保能從 Nginx 輸出中精確捕捉版本差異。

這份完整的清單已準備好，您可以直接將其納入專案的基礎設施維護手冊中。

修訂版本 #1
★由 treeman 建立於 9 ��|�� 2026 16:52:00
由 treeman 更新於 9 ��|�� 2026 16:53:46