**< Kafka Ingress 사용 전환 가이드 >**

**2021 / 06/ 25 우태건**

기본적으로 Ingress Controller의 설정은 <https://github.com/tmax-cloud/install-ingress/tree/5.0> 를 바탕으로 한다. ingress-nginx-system 에 설치된 ingress controller의 설정을 고친다.

설정을 마치고 나면, ingress\_controller\_external\_ip:9001, ingress\_controller\_external\_ip:9002, ingress\_controller\_external\_ip:9003 으로 외부에서 External Sub 가능함.

1. ConfigMap 생성

apiVersion: v1

kind: ConfigMap

metadata:

name: tcp-services

namespace: ingress-nginx-system

labels:

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/part-of: ingress-nginx-system

data:

9001: "hyperauth/kafka-1:9093"

9002: "hyperauth/kafka-2:9093"

9003: "hyperauth/kafka-3:9093"

1. Ingress Controller의 Service 및 Deployment 재생성

apiVersion: v1

kind: Service

metadata:

labels:

helm.sh/chart: ingress-nginx-system-2.10.0

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/instance: ingress-nginx-system

app.kubernetes.io/version: 0.33.0

app.kubernetes.io/managed-by: Helm

app.kubernetes.io/component: controller

name: ingress-nginx-system-controller

namespace: ingress-nginx-system

spec:

type: LoadBalancer

externalTrafficPolicy: Local

ports:

- name: http

port: 80

protocol: TCP

targetPort: http

- name: https

port: 443

protocol: TCP

targetPort: https

**- name: kafka-1**

**port: 9001**

**protocol: TCP**

**- name: kafka-2**

**port: 9002**

**protocol: TCP**

**- name: kafka-3**

**port: 9003**

**protocol: TCP**

selector:

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/instance: ingress-nginx-system

app.kubernetes.io/component: controller

---

apiVersion: apps/v1

kind: Deployment

metadata:

labels:

helm.sh/chart: ingress-nginx-system-2.10.0

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/instance: ingress-nginx-system

app.kubernetes.io/version: 0.33.0

app.kubernetes.io/managed-by: Helm

app.kubernetes.io/component: controller

name: ingress-nginx-system-controller

namespace: ingress-nginx-system

spec:

selector:

matchLabels:

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/instance: ingress-nginx-system

app.kubernetes.io/component: controller

revisionHistoryLimit: 10

minReadySeconds: 0

template:

metadata:

labels:

app.kubernetes.io/name: ingress-nginx-system

app.kubernetes.io/instance: ingress-nginx-system

app.kubernetes.io/component: controller

spec:

dnsPolicy: ClusterFirst

containers:

- name: controller

image: quay.io/kubernetes-ingress-controller/nginx-ingress-controller:0.33.0

imagePullPolicy: IfNotPresent

lifecycle:

preStop:

exec:

command:

- /wait-shutdown

args:

- /nginx-ingress-controller

- --publish-service=ingress-nginx-system/ingress-nginx-system-controller

- --election-id=ingress-controller-leader

- --ingress-class=nginx-system

- --configmap=ingress-nginx-system/ingress-nginx-system-controller

- --validating-webhook=:8443

- --validating-webhook-certificate=/usr/local/certificates/cert

- --validating-webhook-key=/usr/local/certificates/key

**- --tcp-services-configmap=ingress-nginx-system/tcp-services**

securityContext:

capabilities:

drop:

- ALL

add:

- NET\_BIND\_SERVICE

runAsUser: 101

allowPrivilegeEscalation: true

env:

- name: POD\_NAME

valueFrom:

fieldRef:

fieldPath: metadata.name

- name: POD\_NAMESPACE

valueFrom:

fieldRef:

fieldPath: metadata.namespace

livenessProbe:

httpGet:

path: /healthz

port: 10254

scheme: HTTP

initialDelaySeconds: 10

periodSeconds: 10

timeoutSeconds: 1

successThreshold: 1

failureThreshold: 3

readinessProbe:

httpGet:

path: /healthz

port: 10254

scheme: HTTP

initialDelaySeconds: 10

periodSeconds: 10

timeoutSeconds: 1

successThreshold: 1

failureThreshold: 3

ports:

- name: http

containerPort: 80

protocol: TCP

- name: https

containerPort: 443

protocol: TCP

- name: webhook

containerPort: 8443

protocol: TCP

volumeMounts:

- name: webhook-cert

mountPath: /usr/local/certificates/

readOnly: true

resources:

requests:

cpu: 100m

memory: 90Mi

limits:

cpu: 200m

memory: 180Mi

serviceAccountName: ingress-nginx-system

terminationGracePeriodSeconds: 300

volumes:

- name: webhook-cert

secret:

secretName: ingress-nginx-system-admission

---

1. Kafka 인증서 재생성 및 kafka-jks secret 재생성

$ kubectl delete secret kafka-jks -n hyperauth

$ sudo keytool -keystore kafka.broker.truststore.jks -alias ca-cert -import -file /etc/kubernetes/pki/hypercloud-root-ca.crt -storepass tmax@23 -noprompt

$ sudo keytool -keystore kafka.broker.keystore.jks -alias broker -validity 3650 -genkey -keyalg RSA -dname "CN=kafka" -storepass tmax@23 -keypass tmax@23

$ sudo keytool -keystore kafka.broker.keystore.jks -alias broker -certreq -file ca-request-broker -storepass tmax@23

$ cat > "kafka.cnf" <<EOL[kafka]subjectAltName = DNS:kafka-1.hyperauth,DNS:kafka-2.hyperauth,DNS:kafka-3.hyperauth,IP:{NGINX\_INGRESS\_CONTROLLER\_EXTERNAL\_IP}EOL

// ex) NGINX\_INGRESS\_CONTROLLER\_EXTERNAL\_IP = 172.22.6.13, NGINX\_INGRESS\_CONTROLLER\_EXTERNAL\_DNS = 172.22.6.13.nip.io 혹은 Public DNS

$ sudo openssl x509 -req -CA /etc/kubernetes/pki/hypercloud-root-ca.crt -CAkey /etc/kubernetes/pki/hypercloud-root-ca.key -in ca-request-broker -out ca-signed-broker -days 3650 -CAcreateserial -extfile "kafka.cnf" -extensions kafka -sha256

$ sudo keytool -keystore kafka.broker.keystore.jks -alias ca-cert -import -file /etc/kubernetes/pki/hypercloud-root-ca.crt -storepass tmax@23 -noprompt

$ sudo keytool -keystore kafka.broker.keystore.jks -alias broker -import -file ca-signed-broker -storepass tmax@23 -noprompt

$ kubectl create secret generic kafka-jks --from-file=./kafka.broker.keystore.jks --from-file=./kafka.broker.truststore.jks -n hyperauth

1. Kafka Deployment 재생성 ( outside listener 수정 )

apiVersion: apps/v1

kind: Deployment

metadata:

name: kafka-1

namespace: hyperauth

labels:

app: kafka-1

spec:

replicas: 1

selector:

matchLabels:

app: kafka-1

template:

metadata:

labels:

app: kafka-1

spec:

containers:

- name: kafka

image: wurstmeister/kafka:{KAFKA\_VERSION} #2.12-2.0.1

env:

**- name: KAFKA\_ADVERTISED\_LISTENERS**

**value: "INSIDE://kafka-1.hyperauth:9092,OUTSIDE://{ingress\_controller\_external\_ip}:9001"**

**- name: KAFKA\_LISTENERS**

**value: "INSIDE://:9092,OUTSIDE://:9093"**

- name: KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP

value: "INSIDE:SSL,OUTSIDE:SSL"

- name: KAFKA\_INTER\_BROKER\_LISTENER\_NAME

value: "INSIDE"

- name: KAFKA\_ZOOKEEPER\_CONNECT

value: "zookeeper:2181"

- name: KAFKA\_SSL\_KEYSTORE\_LOCATION

value: "/certs/kafka.broker.keystore.jks"

- name: KAFKA\_SSL\_KEYSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_KEY\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_TRUSTSTORE\_LOCATION

value: "/certs/kafka.broker.truststore.jks"

- name: KAFKA\_SSL\_TRUSTSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_ENDPOINT\_IDENTIFICATION\_ALGORITHM

value: ""

- name: KAFKA\_CREATE\_TOPICS

value: "tmax:1:3" # Topic명:Partition개수:Replica개수

- name: KAFKA\_BROKER\_ID

value: "501"

- name: KAFKA\_AUTO\_CREATE\_TOPICS\_ENABLE

value: "true"

- name: KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR

value: "3"

- name: KAFKA\_LOG\_DIRS

value: "/kafka/logs"

ports:

- name: client

containerPort: 9092

volumeMounts:

- name: kafka-data

mountPath: /kafka

- name: ssl

mountPath: /certs

resources:

limits:

cpu: "1"

memory: "1Gi"

requests:

cpu: "1"

memory: "1Gi"

volumes:

- name: kafka-data

persistentVolumeClaim:

claimName: kafka-pvc-1

- name: ssl

secret:

secretName: kafka-jks

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: kafka-2

namespace: hyperauth

labels:

app: kafka-2

spec:

replicas: 1

selector:

matchLabels:

app: kafka-2

template:

metadata:

labels:

app: kafka-2

spec:

containers:

- name: kafka

image: wurstmeister/kafka:{KAFKA\_VERSION} #2.12-2.0.1

env:

**- name: KAFKA\_ADVERTISED\_LISTENERS**

**value: "INSIDE://kafka-1.hyperauth:9092,OUTSIDE://{ingress\_controller\_external\_ip}:9002"**

**- name: KAFKA\_LISTENERS**

**value: "INSIDE://:9092,OUTSIDE://:9093"**

- name: KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP

value: "INSIDE:SSL,OUTSIDE:SSL"

- name: KAFKA\_INTER\_BROKER\_LISTENER\_NAME

value: "INSIDE"

- name: KAFKA\_ZOOKEEPER\_CONNECT

value: "zookeeper:2181"

- name: KAFKA\_SSL\_KEYSTORE\_LOCATION

value: "/certs/kafka.broker.keystore.jks"

- name: KAFKA\_SSL\_KEYSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_KEY\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_TRUSTSTORE\_LOCATION

value: "/certs/kafka.broker.truststore.jks"

- name: KAFKA\_SSL\_TRUSTSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_ENDPOINT\_IDENTIFICATION\_ALGORITHM

value: ""

- name: KAFKA\_CREATE\_TOPICS

value: "tmax:1:3" # Topic명:Partition개수:Replica개수

- name: KAFKA\_AUTO\_CREATE\_TOPICS\_ENABLE

value: "true"

- name: KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR

value: "3"

- name: KAFKA\_LOG\_DIRS

value: "/kafka/logs"

- name: KAFKA\_BROKER\_ID

value: "502"

ports:

- name: client

containerPort: 9092

volumeMounts:

- name: kafka-data

mountPath: /kafka

- name: ssl

mountPath: /certs

resources:

limits:

cpu: "1"

memory: "1Gi"

requests:

cpu: "1"

memory: "1Gi"

volumes:

- name: kafka-data

persistentVolumeClaim:

claimName: kafka-pvc-2

- name: ssl

secret:

secretName: kafka-jks

---

apiVersion: apps/v1

kind: Deployment

metadata:

name: kafka-3

namespace: hyperauth

labels:

app: kafka-3

spec:

replicas: 1

selector:

matchLabels:

app: kafka-3

template:

metadata:

labels:

app: kafka-3

spec:

containers:

- name: kafka

image: wurstmeister/kafka:{KAFKA\_VERSION} #2.12-2.0.1

env:

**- name: KAFKA\_ADVERTISED\_LISTENERS**

**value: "INSIDE://kafka-1.hyperauth:9092,OUTSIDE://{ingress\_controller\_external\_ip}:9003"**

**- name: KAFKA\_LISTENERS**

**value: "INSIDE://:9092,OUTSIDE://:9093"**

- name: KAFKA\_LISTENER\_SECURITY\_PROTOCOL\_MAP

value: "INSIDE:SSL,OUTSIDE:SSL"

- name: KAFKA\_INTER\_BROKER\_LISTENER\_NAME

value: "INSIDE"

- name: KAFKA\_ZOOKEEPER\_CONNECT

value: "zookeeper:2181"

- name: KAFKA\_SSL\_KEYSTORE\_LOCATION

value: "/certs/kafka.broker.keystore.jks"

- name: KAFKA\_SSL\_KEYSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_KEY\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_TRUSTSTORE\_LOCATION

value: "/certs/kafka.broker.truststore.jks"

- name: KAFKA\_SSL\_TRUSTSTORE\_PASSWORD

valueFrom:

secretKeyRef:

name: passwords

key: CERTS\_PASSWORD

- name: KAFKA\_SSL\_ENDPOINT\_IDENTIFICATION\_ALGORITHM

value: ""

- name: KAFKA\_CREATE\_TOPICS

value: "tmax:1:3" # Topic명:Partition개수:Replica개수

- name: KAFKA\_AUTO\_CREATE\_TOPICS\_ENABLE

value: "true"

- name: KAFKA\_OFFSETS\_TOPIC\_REPLICATION\_FACTOR

value: "3"

- name: KAFKA\_LOG\_DIRS

value: "/kafka/logs"

- name: KAFKA\_BROKER\_ID

value: "503"

ports:

- name: client

containerPort: 9092

volumeMounts:

- name: kafka-data

mountPath: /kafka

- name: ssl

mountPath: /certs

resources:

limits:

cpu: "1"

memory: "1Gi"

requests:

cpu: "1"

memory: "1Gi"

volumes:

- name: kafka-data

persistentVolumeClaim:

claimName: kafka-pvc-3

- name: ssl

secret:

secretName: kafka-jks