SNMP feed implementation:-  
  
Overview of SNMP 3rd-Party Feed Implementation (Prod Environment)

You're integrating SNMP feeds into your system so alerts can be sent as SNMP traps to a customer-defined destination. This enables external systems (likely at the customer’s end) to receive and act on alerts generated within your platform.

🔧 Key Components & Configuration

✅ 1. Notification Microservice

* **Image Used:** gcr.io/quixotic-skill-272111/notificationservice\_dev:latest
* **Configuration Added:**
* alert.snmp.receiverIp=10.93.2.194: Customer's IP to receive traps.
* alert.snmp.receiverPort=31524: Destination port for SNMP traps.
* alert.snmp.receiverName=bsnl: Logical identifier for the customer or context.

This setup ensures that SNMP traps can be dispatched to the specified endpoint when an alert is triggered.

✅ 2. Alert Engine Microservice

* **Image Used:** gcr.io/quixotic-skill-272111/optigoalertengine\_bsnl:7.0.0.27
* **Configuration Added:**
* snmp.topic=snmpNotify: SNMP notification topic in Kafka.
* kafka.topicList=emailNotify,snmpNotify: Defines messaging pipelines for alert dispatch.

This allows the alert engine to publish SNMP-related events to Kafka, which the notification service can consume and convert into actual SNMP traps sent to the customer.

🔁 End-to-End Flow Summary

1. **Alert Engine** detects a qualifying alert and publishes it to snmpNotify topic in Kafka.
2. **Notification Microservice** picks up the event from Kafka and formats it into an SNMP trap.
3. The trap is sent over the network to the customer-defined IP and port (10.93.2.194:31524).
4. **Customer's SNMP Listener** receives the trap and processes it per their operational needs.

🛡️ Best Practices You’re Following

* **Explicit versioning and isolation** of microservice images for production stability.
* **Property-driven configuration** that allows easy tuning without code changes.
* **Kafka integration** for scalable and decoupled alert dissemination.
* **Customer alignment** by using IP, port, and SNMP version sourced from them.

**Objective**

Enable BSNL’s alert system to send SNMP traps to an external destination defined by the customer, facilitating real-time monitoring on their end.

🔧 Configuration Details

**Customer Inputs Received:**

* **Receiver IP:** 10.93.2.194
* **Receiver Port:** 31524
* **SNMP Version:** [Customer-confirmed version]
* **Receiver Name:** bsnl

**Microservice Configurations:**

1. **Notification Microservice**
   * **Image:** gcr.io/quixotic-skill-272111/notificationservice\_dev:latest
   * **Property File Updates:**

alert.snmp.receiverIp=10.93.2.194 alert.snmp.receiverPort=31524 alert.snmp.receiverName=bsnl

 **Alert Engine Microservice**

* **Image:** gcr.io/quixotic-skill-272111/optigoalertengine\_bsnl:7.0.0.27
* **Property File Updates:**

snmp.topic=snmpNotify kafka.topicList=emailNotify,snmpNotify

🔁 Flow of SNMP Trap Delivery

1. **Alert Engine** triggers an alert and publishes the event to the snmpNotify Kafka topic.
2. **Notification Microservice** consumes the Kafka event and formats it as an SNMP trap.
3. The trap is dispatched to the customer endpoint (10.93.2.194:31524) via SNMP protocol.
4. The customer validates trap receipt using their SNMP listener or management platform.

Next Steps

1. **Verification in Production:** Ensure alerts are triggering corresponding SNMP traps.
2. **Monitoring Setup:** Work with the customer to confirm trap reception and interpretation.
3. **Logging & Observability:** Validate logs on both ends to ensure visibility of SNMP