Weather Forecast Data Streaming Project using Big Data tools

Project Overview

This project demonstrates real-time weather forecast data streaming as part of a Big Data Engineering pipeline. Weather data is ingested, processed, and visualized to provide actionable insights. The project simulates real-world data processing scenarios, focusing on data engineering tools and techniques suitable for high-volume data streams.

> *Note: This pipeline was implemented in CentOS 6.5.*

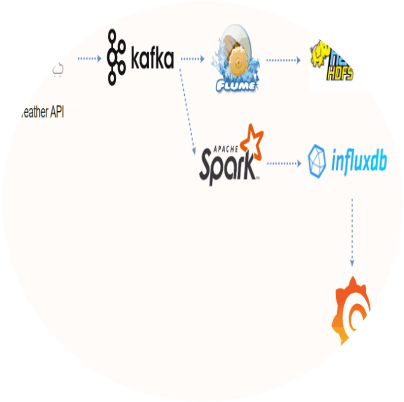
Project Objectives

- Stream real-time weather forecast data to a Kafka topic.
- Process the streaming data using PySpark to transform it for storage and analysis.
- Use InfluxDB for time-series data storage and HDFS for scalable storage.
- Visualize processed data in Grafana to monitor forecast trends.

Key Components

- **Data Source**: Weather forecast data (https://www.weather.gov/documentation/services-web-api) streamed using Kafka.
- **Message Broker**: Apache Kafka for ingesting and streaming data.
- **Data Processing**: PySpark transforms the raw forecast data and prepares it for storage.
- **Storage**:
    - **InfluxDB** for time-series data storage and analysis.
    - **HDFS** for scalable storage of processed data using Apache Flume to transfer from Kafka.
- **Visualization**: Grafana for monitoring forecast trends.

Architecture Diagram



Tools and Technologies

- **Kafka**: For real-time data streaming.
- **Apache Flume**: To transfer data from Kafka to HDFS.
- **PySpark**: For processing and transforming streaming data.
- **InfluxDB**: A time-series database for storing weather data.
- **Grafana**: Visualization tool for monitoring trends.
- **Jupyter Notebook**: Documentation and interactive development environment.

Project Workflow

1. **Data Streaming**:
    - Weather forecast data is streamed from a source (using Python's KafkaProducer) and published to a Kafka topic.
2. **Data Ingestion to HDFS**:
    - Apache Flume is used to efficiently transfer data from Kafka to HDFS, providing a backup storage layer.
3. **Data Processing**:
    - Data is consumed from Kafka using PySpark.
    - Transforms are applied to pivot the data by day and time (morning/night), removing irrelevant entries (e.g., 'Overnight').
4. **Storage and Visualization**:
    - Processed data is saved to InfluxDB, and real-time visualizations are configured in Grafana.

Metadata used:

- Location: Latitude and longtitude are set on Washington, D.C.
- short_forecast: A brief description of the forecast (e.g., "Partly Cloudy").
- Temperature: The forecasted temperature.

- Temperature unit: The unit of temperature (Fahrenheit).
- Wind speed: The forecasted wind speed.
- Start-time: The start time of the forecast period which is every 7 days.

Usage:

Step #1: Run the following commands in separate terminals

Start HDFS and YARN:

```
start-all.sh
```

Start Zookeeper Server:

```
cd $KAFKA_HOME
```

```
bin/zookeeper-server-start.sh config/zookeeper.properties
```

Start Kafka Server:

```
cd $KAFKA_HOME
```

```
bin/kafka-server-start.sh config/server.properties
```

(Optional) Checking ecosystem running successfully:

```
jps
```

Choose either:

▶ Weather API -> Kafka -> Flume -> HDFS

▶ Weather API -> Kafka -> PySpark -> influxDB -> Grafana

Future Improvements

- Enhance data cleaning and transformation processes.
- Add support for additional weather parameters.
- Explore alternative storage options for scalability.