SAKARYA
ÜNİVERSİTESİ

Fall 2025-2026
SWE-423 Data Analytics
Asst. Prof. Nur Banu OĞUR

Real-time Network Intrusion Detection System with Big Data Technologies

Görkem ÇETİNKAYA - B211202022

1. Summary of the Project

This project aims to establish a scalable, real-time data analytics system for Network Intrusion Detection (NIDS). The primary objective is to simulate a streaming architecture where network traffic data is ingested, processed, analyzed, and stored instantaneously using Big Data technologies.

The system is designed to distinguish between "Normal" traffic and "Anomalous" (Attack) traffic. It leverages **Apache Kafka** for message distribution, **Apache Spark** for real-time stream processing and machine learning, and **MongoDB**for raw data persistence. The entire infrastructure is containerized using **Docker** to ensure reproducibility and scalability.

2. Technologies Used

The following technologies were utilized to build the end-to-end pipeline:

- **Apache Kafka & Zookeeper:** Used as the backbone for data ingestion and message distribution. It decouples the data source from the processing engine.
- **Apache Spark (Structured Streaming):** Used for processing the continuous stream of data.
- **Spark MLlib:** Used to build the Machine Learning pipeline (Logistic Regression) for classifying network traffic.
- **MongoDB:** A NoSQL database used to store the raw network traffic data for archival and future auditing.
- **Docker & Docker Compose:** Used to orchestrate the services (Broker, Zookeeper, Spark Master/Worker, MongoDB) in an isolated environment.
- **Python (PySpark, Kafka-Python):** The primary programming language used for Producer, Consumer, and Processor scripts.

3. Data Analysis Methods Used

- **Data Ingestion:** A Producer script simulates real-time traffic by reading the `KDDTest+` dataset and pushing messages to the Kafka topic `network-traffic`.
- **Pre-processing Pipeline:** The raw CSV data streaming from Kafka is parsed into a structured format. Categorical features (Protocol, Service, Flag) are converted into numerical indices using `StringIndexer`. All features are vectorized using `VectorAssembler`.
- **Machine Learning Model:** A Supervised Learning approach is implemented. A **Logistic Regression** model is trained on a static dataset (`KDDTrain+`) to learn attack patterns.
- **Real-time Prediction & Evaluation:** The trained model is applied to the live data stream. The system calculates **Batch Accuracy** (for the current micro-batch) and **Global Accuracy** (cumulative performance) in real-time to monitor system reliability.

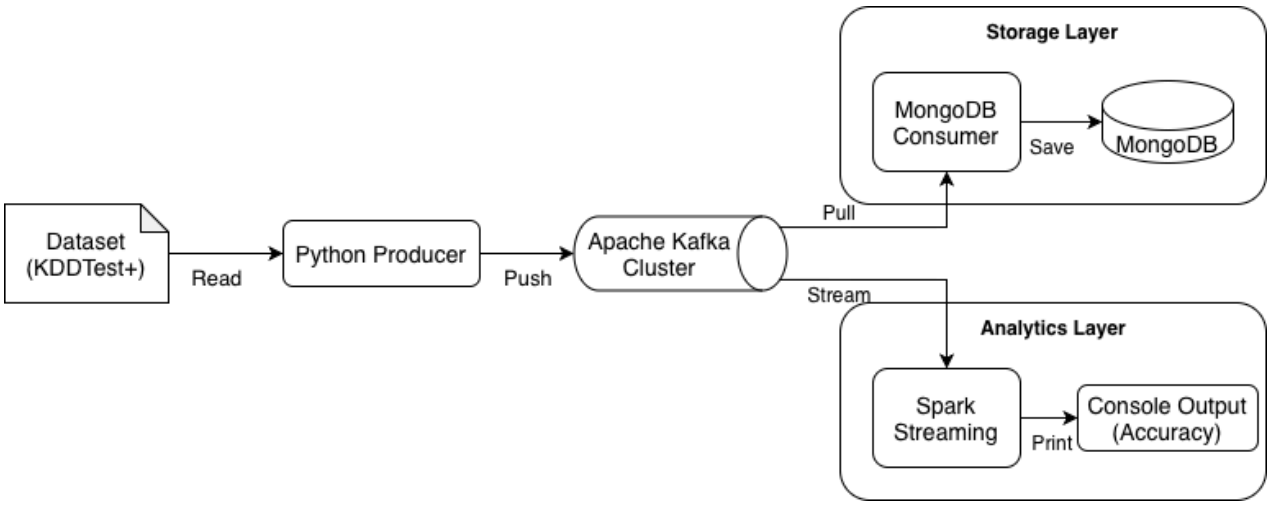
4. Description of the Dataset

The project utilizes the **NSL-KDD** dataset, which is a refined version of the benchmark KDD'99 dataset for intrusion detection.

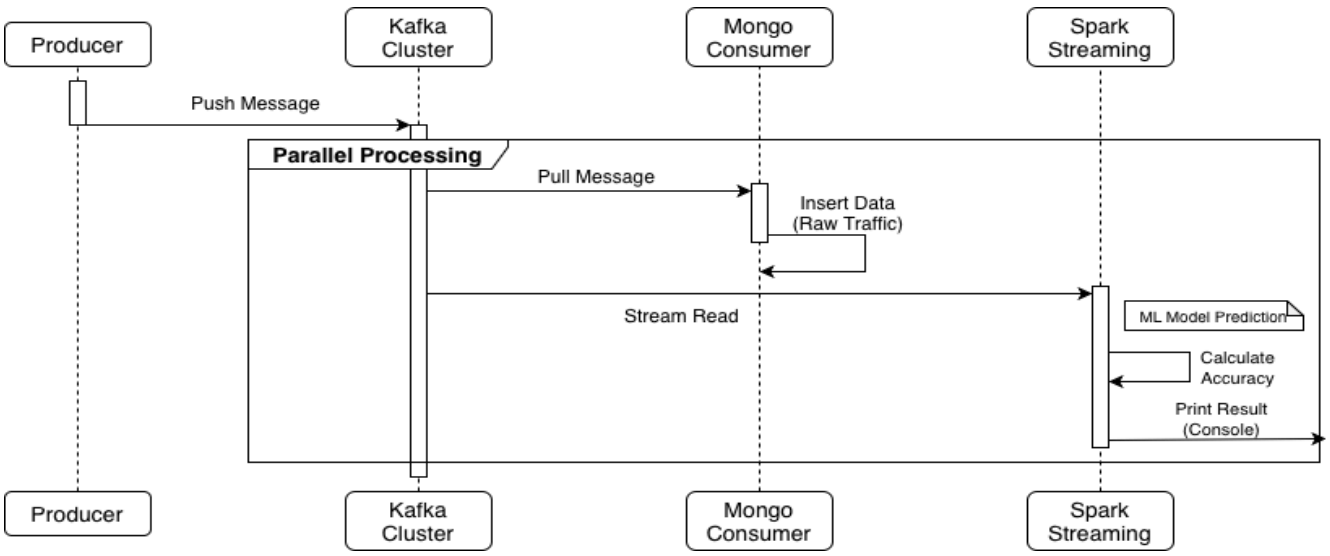
- **Training Set:** `KDDTrain+.txt` was used to train the Logistic Regression model.
- **Testing Set:** `KDDTest+.txt` was used for the streaming simulation to ensure the model is tested on unseen data (preventing data leakage).
- **Features:** The dataset includes basic network features such as `duration`, `protocol_type` (TCP/UDP/ICMP), `service`(HTTP, FTP, etc.), `src_bytes`, and `dst_bytes`.
- **Labels:** The traffic is labeled as either `normal` or specific attack types (e.g., `neptune`, `satan`, `smurf`).

5. General Flow Chart of the Project

The architecture follows a Producer-Consumer model integrated with a Stream Processor.



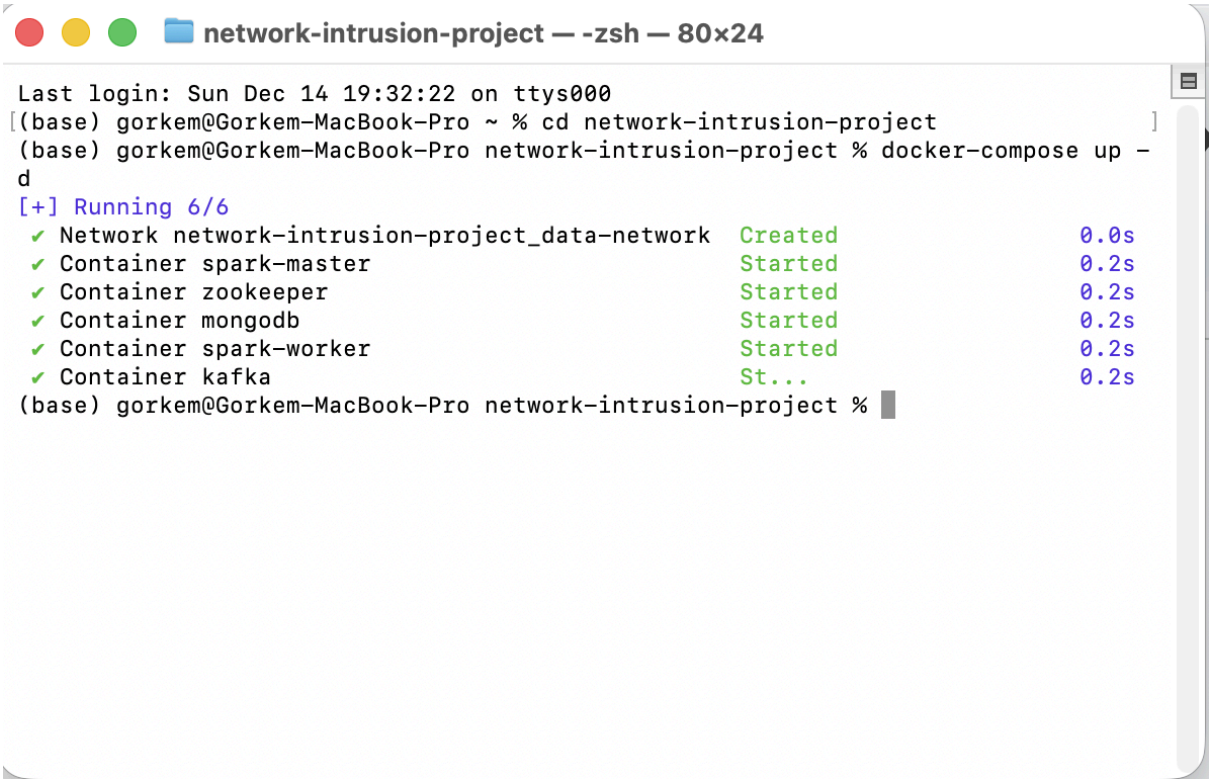
6. Timing Chart (Sequence Diagram)



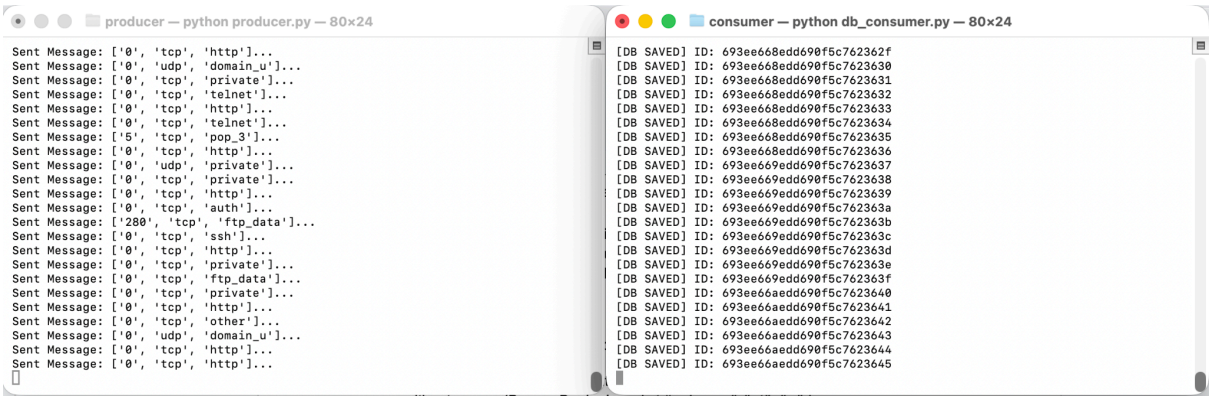
7. Findings Obtained

The system was successfully deployed and tested. The following findings were obtained:

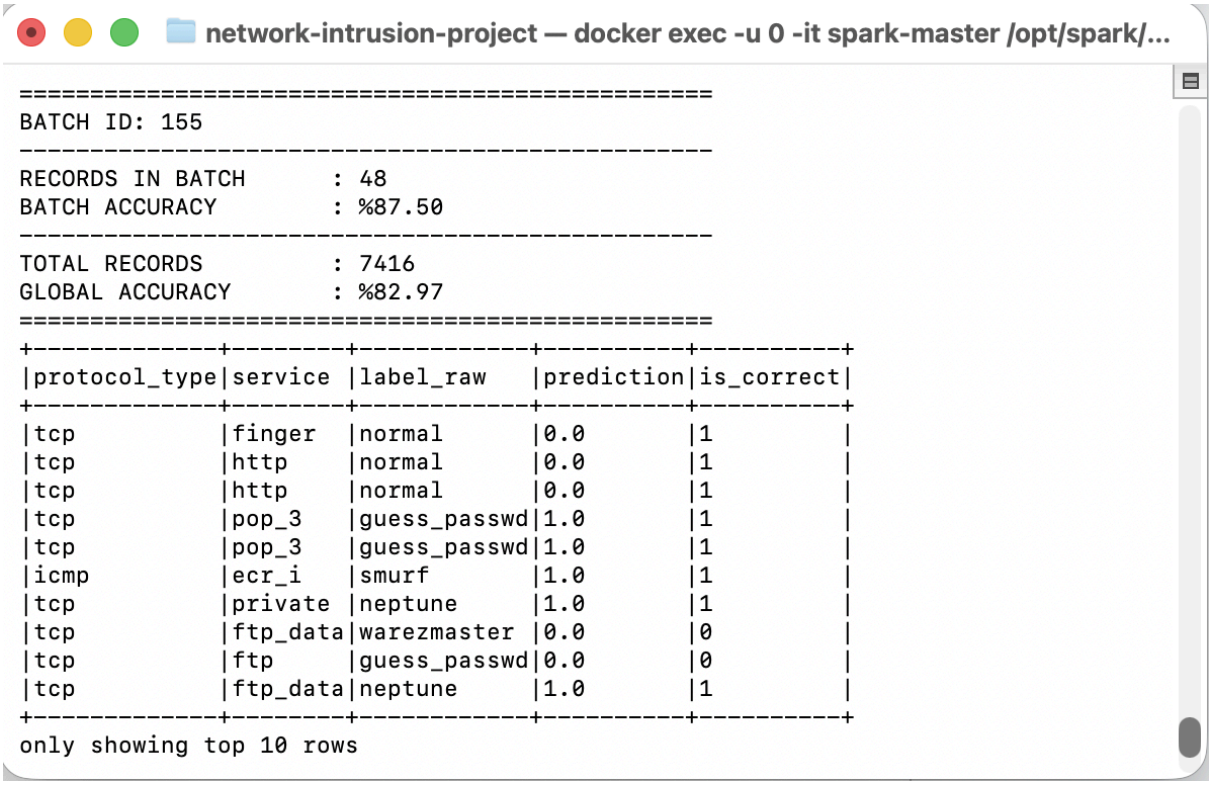
- **Infrastructure:** All Docker containers (Zookeeper, Kafka, Spark, MongoDB) operated without errors.



- **Data Streaming & Persistence:** The Producer successfully streamed data, and the Consumer stored over 36,000 records to MongoDB, proving the system's data integrity.



- **Real-time Analytics:** The Spark engine successfully processed micro-batches. The Logistic Regression model achieved a **Global Accuracy of ~82.97%** on the test stream. The model correctly identified various attack types (e.g., `neptune`, `warezmaster`) and distinguished them from normal traffic.



- **Database Verification:** Verification scripts confirmed that data is actively being written to MongoDB.