# Architecture Document for DDoS Detection Research Project

## 1. Application Architecture

### 1.1 Microservices

The architecture is composed of distinct microservices that independently handle various aspects of the DDoS detection system. Each service is responsible for specific functionality:

**Preprocessing Service:** Manages data cleaning, feature selection, and data transformation tasks.

**Model Training Service:** Trains models such as KNN, CNN, and H-QNN on the processed dataset.

**Prediction Service:** Provides real-time DDoS traffic classification by deploying trained models to predict network traffic behaviour.

**Monitoring Service:** Monitors system performance, detects anomalies, and raises alerts for detected DDoS threats.

### 1.2 Event-Driven

The system is event-driven, reacting to triggers such as new incoming network traffic data or model performance thresholds. Events include:

**Data Ingestion:** When new traffic data is collected, it triggers preprocessing and storage services.

**Model Retraining:** An event is triggered if model accuracy or performance drops below a certain threshold, prompting the system to retrain the models.

**DDoS Detection Event:** When the system detects abnormal traffic behavior (potential DDoS attack), an alert is generated, triggering a mitigation response.

### 1.3 Serverless

Some components, such as the preprocessing tasks and certain low-latency API calls, are designed to run serverless for scalability. For instance:

**Serverless Functions:** These handle smaller tasks like preprocessing data batches, responding to API calls for real-time classification, and triggering retraining.

**Quantum Computing Integration:** For hybrid quantum model (H-QNN) computations, serverless cloud-based quantum computing platforms (e.g., IBM Quantum or AWS Braket) are integrated for optimal performance.

# 2. Database Architecture

## 2.1 ER Diagram

The database will store network traffic data, model configurations, and detection outcomes. The key entities are:

**Network Traffic:** Stores the traffic data features like IP, packet size, protocol, and timestamp.

**Models:** Contains details of the trained models (KNN, CNN, H-QNN) such as model weights, hyperparameters, and training logs.

**Detection Logs:** Logs the results of DDoS detection, including predictions, timestamps, and confidence scores.

**Alerts:** Stores information about triggered DDoS alerts, including the type of attack and its severity.

## 2.2 Schema Design

The schema follows a relational model:

Traffic_Data (id, source_ip, dest_ip, packet_size, protocol, timestamp)

Models (id, model_type, model_config, accuracy, last_trained)

Detection_Log (id, traffic_id, prediction, confidence, detected_at)

Alerts (id, detection_log_id, alert_type, severity, generated_at)

# 3 Data Exchange Contract

## 3.1 Frequency of Data Exchanges

**Training Data Exchange:** The training dataset (CIC-DDoS2019) is processed in batch mode, and models are retrained periodically based on new data or performance drops.

**Real-time Data Exchange:** Incoming network traffic is processed continuously, with predictions made in real time to detect DDoS attacks.

## 3.2 Data Sets

**CIC-DDoS2019 Dataset:** Used for initial model training, containing a wide range of DDoS attacks.

**Real-time Traffic Data:** Incoming data from live network traffic for real-time DDoS detection.

## 3.3 Mode of Exchanges

**API:** Traffic data is ingested via REST APIs, which then feed the data to the appropriate preprocessing and detection services.

**Queue:** A message queue (e.g., RabbitMQ) is used to handle incoming traffic bursts and distribute the data to the prediction or preprocessing services.

**File Exchange:** Model training jobs use batch data files (e.g., CSV) for training and validation.