

GAOUI Yanis, HUMEAU Gauthier, BOIZET Hugo, DESMONS Damien, PITRAT Clement, FAVRE Hugo

# REAL-TIME FRAUD DETECTION

# THE PROBLEM

- Card fraud losses : \$32 billion (2023)
- Thousands of transactions per second
- Detect BEFORE completion (< 5 sec)
- Minimize false positives

# REQUIREMENTS

- Real-time detection
- Handle late events
- Handle duplicates
- ML Fraud scoring
- Dashboard

# LAMBDA ARCHITECTURE

## Speed Layer

- Spark Streaming
- Real-time
- 2-3 sec latency

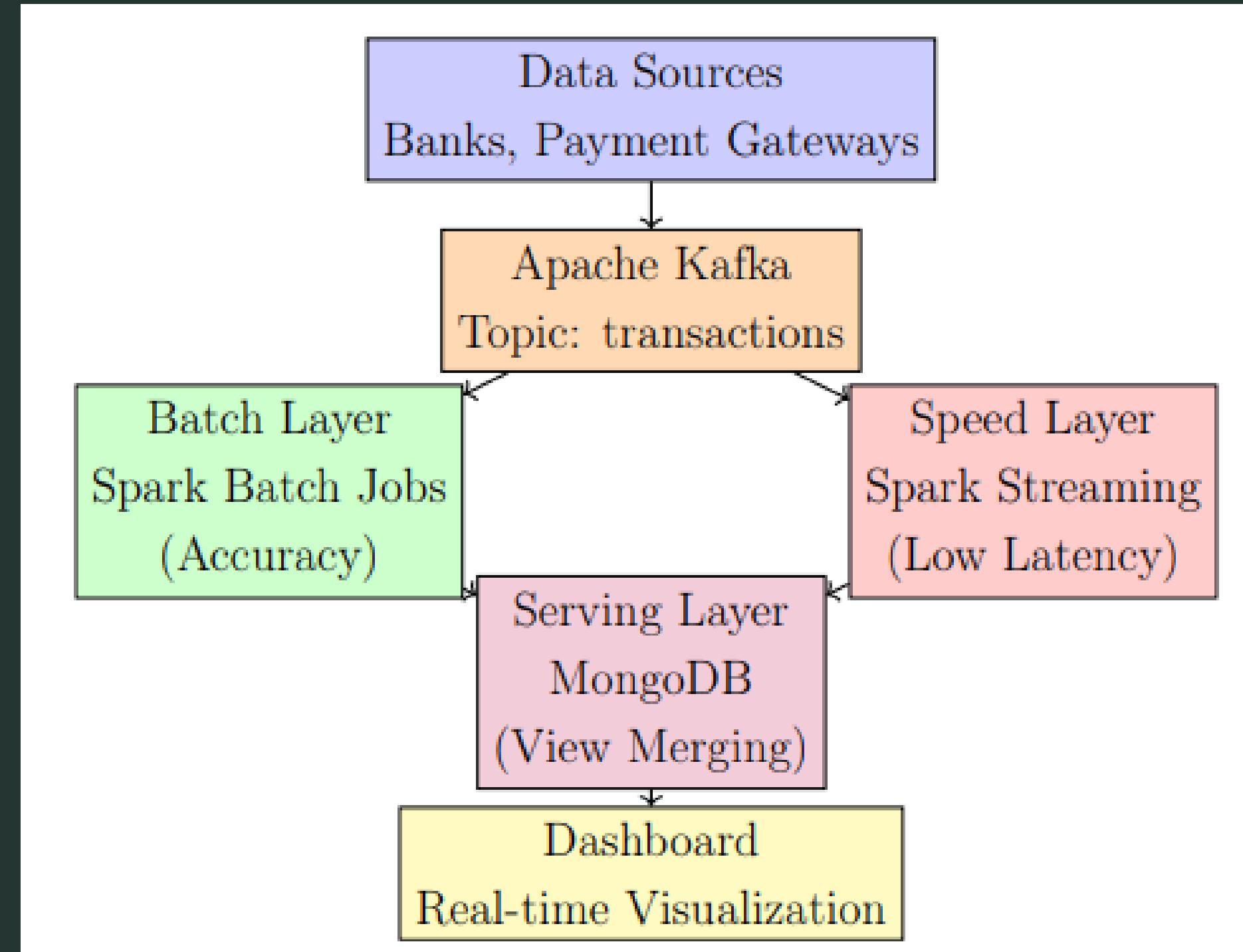
## Batch Layer

- Spark Batch
- Historical data
- 100% accuracy

## Batch Processing

- MongoDB
- Fast and Reliable

# LAMBDA ARCHITECTURE



# WHY BIG DATA ARCHITECTURE ?

- **Volume** : Billions of Transactions/day → Spark Distributed
- **Velocity** : Thousands TPS → Kafka + Streaming
- **Variety** : Transaction + User + Merchant → Flexible Mongo
- **Veracity** : Duplicates, Late Events → Watermark, Deduplicates
- **Value** : Losses → Real Time Detection

# LAMBDA VS KAPPA

Aspect	Lambda	Kappa
Accuracy	Complete recomputation	May drift
ML Training	Batch on all data	Complex
Audit trail	Native	Requires replay
Complexity	2 codebases	1 codebase

# TECHNOLOGY STACK

- **Kafka** : High-performance message broker
- **Apache Spark** : Batch + Streaming processing
- **MongoDB** : Serving layer database
- **Docker + Kubernetes** : Orchestration + auto-scaling
- **Grafana / Dash** : Monitoring

# TECHNOLOGY STACK

Component	Technology
Message Broker	<ul style="list-style-type: none"><li>Apache Kafka</li><li>6 partitions</li><li>7-day retention</li></ul>
Batch Processing	<ul style="list-style-type: none"><li>Apache Spark (Batch)</li><li>MLlib for ML</li><li>Window function</li></ul>
Stream Processing	<ul style="list-style-type: none"><li>Spark Structured Streaming</li><li>Watermarking</li><li>Checkpointing</li></ul>
Serving Database	<ul style="list-style-type: none"><li>MongoDB</li><li>Flexible schema</li><li>Aggregation pipeline</li></ul>
Infrastructure	Docker + Kubernetes
Monitoring	Prometheus + Grafana

# KAFKA CONFIGURATION

- 6 Partitions → Parallel Consumption + user\_id Key
- 3 Replication Factors → Survive Broker Failure + No Data Loss
- 7 Retention Days → Speed Layer Reprocessing
- Producer → Waits for all + Unique Semantics

# ML + FRAUD SCORING ALGORITHM

**Score =**

$$0.25 \times \text{Country} + 0.20 \times \text{Category} + 0.20 \times \text{Amount} + 0.10 \times \text{Online} + 0.10 \times \text{Hour} + 0.15 \times \text{Velocity}$$

Alert threshold: Score  $\geq 0.6$

High-risk countries : 0.7-0.8

Crypto transactions : 0.85-0.9

# REAL-TIME DASHBOARD

- 4 KPIs : Total transactions, Fraud count, Fraud rate
- Real-time chart (transactions/hour)
- World map (fraud by country)
- Breakdown by merchant category
- Fraud alert history

# KEY RESULTS

**2-3 sec**

Latency

**95%**

Accuracy

**10 TPS**

Throughput

# FUTURE IMPROVEMENTS

- Cloud deployment (AWS/GCP)
- Advanced ML models (XGBoost, LSTM)
- Feature Store (Feast/Tecton)
- A/B testing for model deployment

# DEMONSTRATION OF OUR PROJECT

# THANKS FOR YOUR ATTENTION

Any questions about our Project ?