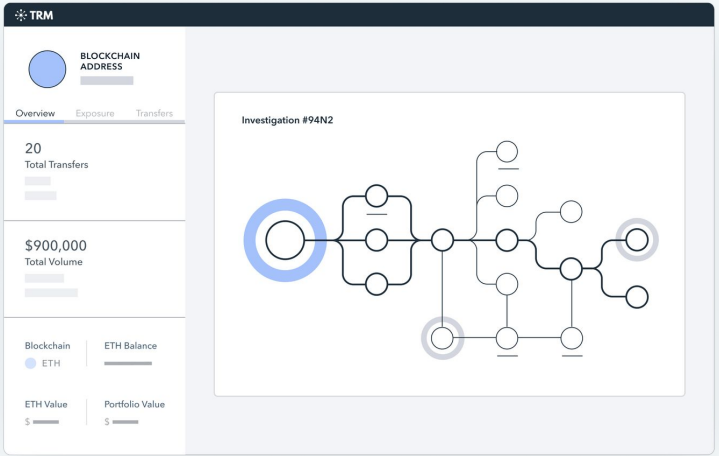
# Architecting Real-Time Blockchain Intelligence with Apache Beam and Apache Kafka

Vijay Shekhawat,
Staff Software Engineer, TRM Labs

BEAM SUMMIT
NYC 2025

# About TRM Labs

- TRM Labs provide blockchain intelligence tools to help financial institutions, crypto businesses, and government agencies detect and investigate crypto-related financial crime and fraud.
- TRM's Analytics platform processes petabytes of blockchain data across 60+ blockchains and answers hundreds of concurrent queries.



# Agenda

1. Real-Time Blockchain Intelligence That Stops Crime
2. Exploring Stream Processing Architectures
3. Why Apache Beam + Dataflow as the stream processing engine
4. Design principles & Performance optimizations
5. Multi-cloud stream processing with Apache Beam
6. Practical Recommendations for Adopting Stream Processing

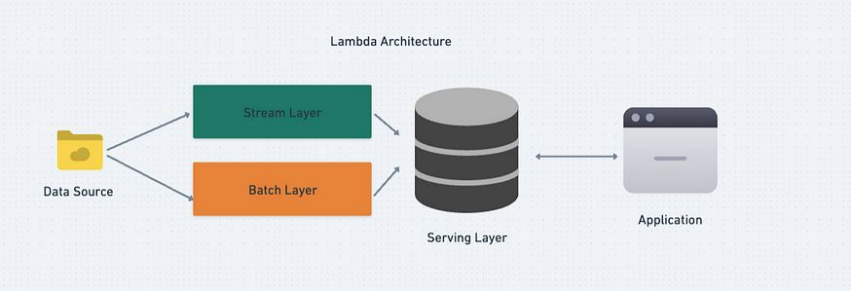
# Real-Time Blockchain Intelligence That Stops Crime

1. Follow the money in real time — across wallets, chains, and cash-out points
2. Critical insights that enable investigators to stop financial crime before it's too late
3. Always-on coverage: blockchain activity never sleeps — 24/7, 365 days a year
4. Processing tens of thousands of blockchain events per second across 60+ chains
5. Built for speed, scale, and reliability — because real people are counting on it

# Exploring Stream Processing Architectures

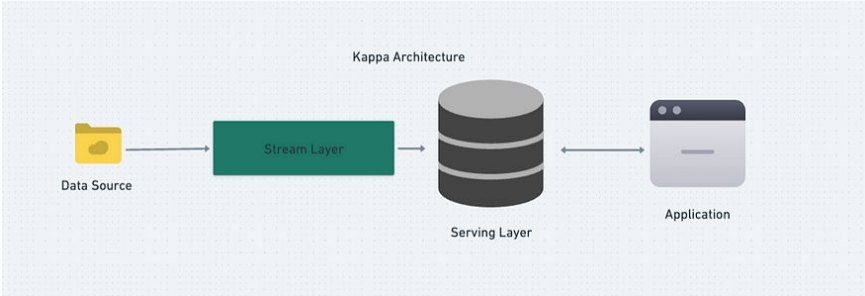
1. A way manage ingestion, processing, and storage of large-scale data
2. At TRM, we evaluated:
    - Lambda architecture – supports batch + real-time processing
    - Kappa architecture – focuses on real-time stream processing
3. Each model has distinct strengths and trade-offs depending on the workload

# Lambda Architecture




- Dual-layer approach: Batch and Stream.
- Robust, fault-tolerant, suitable for complex tasks.
- Complexity in maintaining two systems.

# Kappa Architecture




- Single stream-processing layer.
- Simpler, easier maintenance.
- Less efficient for complex processing.

# Why Apache Beam + Dataflow as the stream processing engine

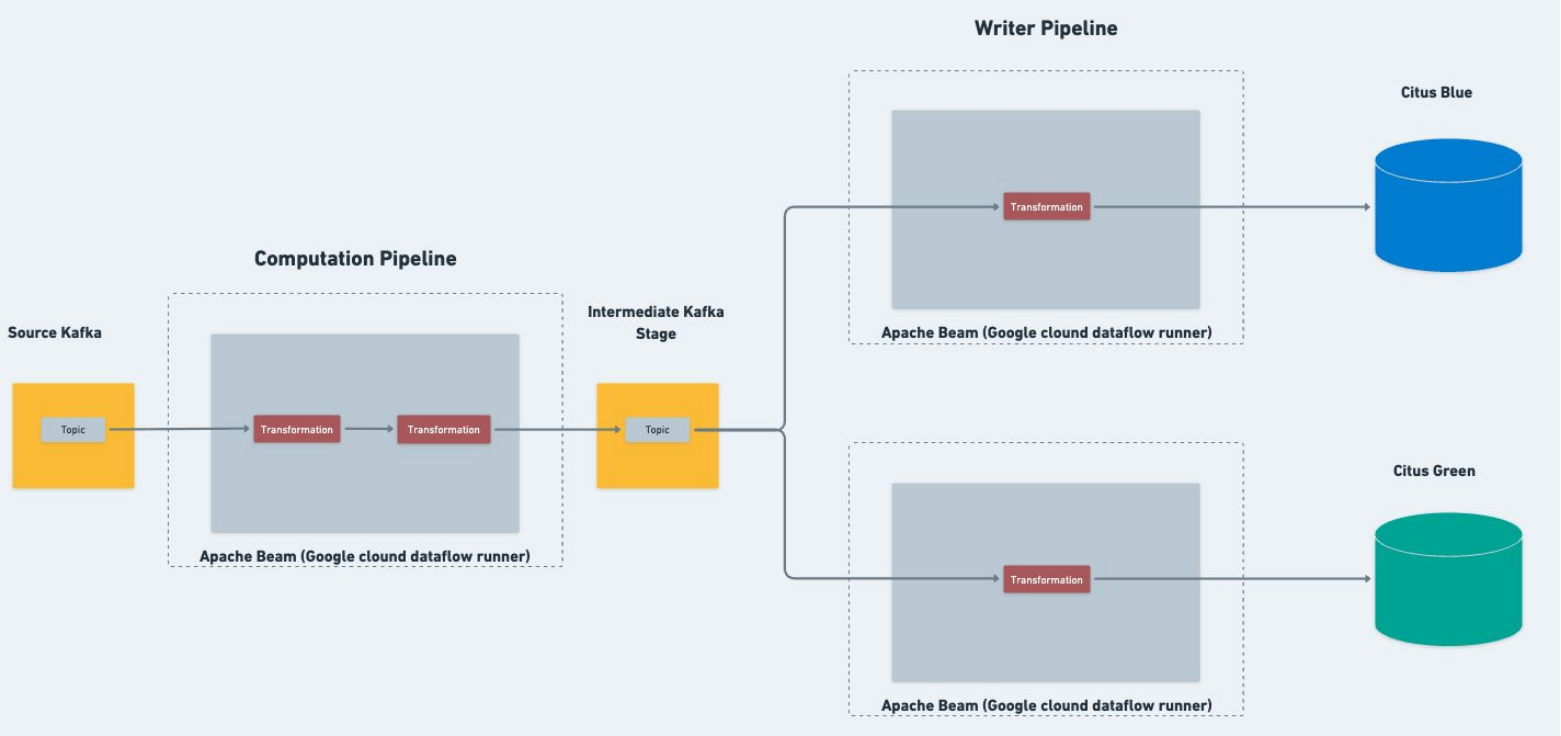
1. Define the evaluation criteria
    - Performance & Scalability
    - Maintenance Overhead
    - System Compatibility
    - Cost Efficiency
2. Know your strengths and weaknesses
    - Team Expertise
    - Future goals
3. Compare options
    - Apache Beam
    - Spark Streaming
    - Apache Flink
    - Kafka Streams

# Why Apache Beam + Dataflow as the stream processing engine

| Criteria | Apache Beam with GCP Dataflow | Spark Structured Streaming | Kafka Streams | Apache Flink |
|---|---|---|---|---|
| **Performance** | A | A | B | A+ |
| **Scalability** | A+ | A | C | A |
| **Maintenance Overhead** | A+ | C | B | C |
| **Compatibility** | A+ | B | B | B |
| **Cost** | B+ | B | C | B |

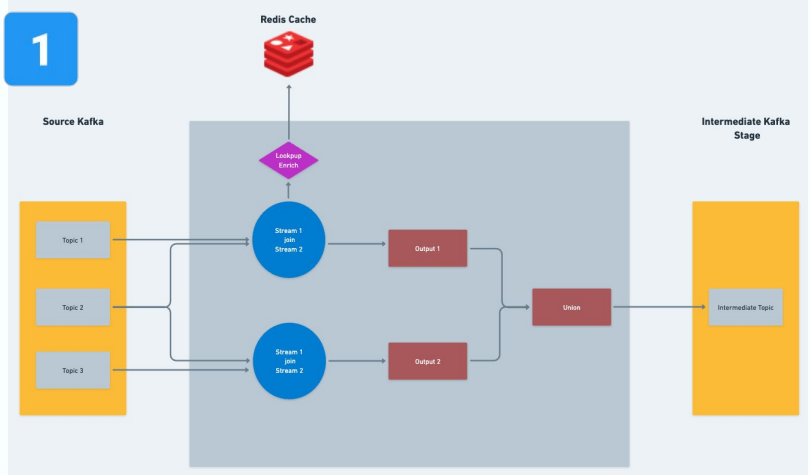
# Design Principles for Realtime stream processing at TRM Labs

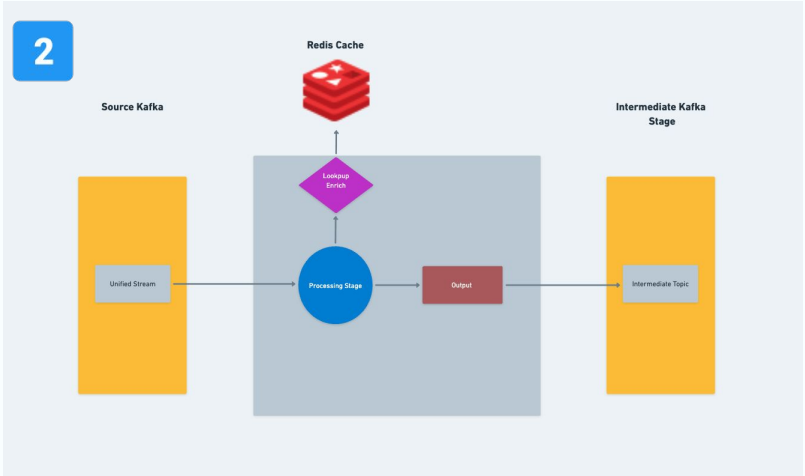
# Design Principle – Separate Computational From Write Stages



# Design Principle – Avoid Streaming joins

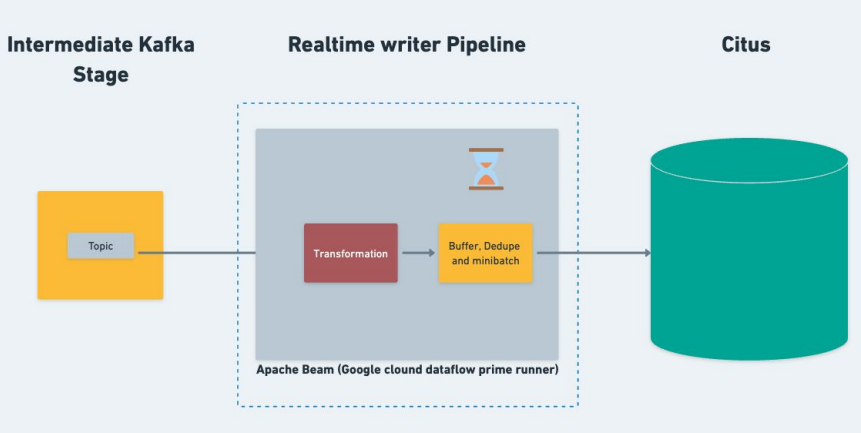
- Initial Challenges with V1:
  - Managing Out-of-Order events across multiple streams.
  - Strategic windowing, watermarks, triggers for accurate data joins.
- Evolution:
  - Moved from stream joins to a Unified Stream Approach.
  - Achieved 100% data accuracy.
  - Considered granularity, data constituents, Kafka topic partition optimization.



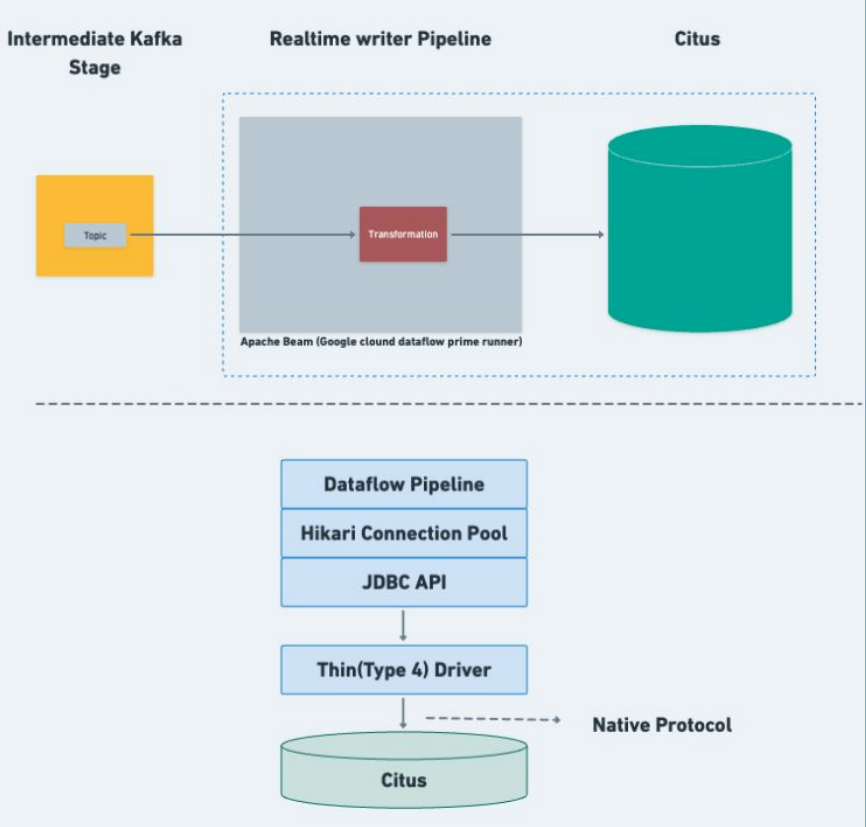
# Design Principle – Layered approach

- **Layer 1: Pipeline Configuration**
  - **Batched JDBC writes.**
  - **Buffering via Apache Beam windows for deduplication.**
- Layer 2: Network Configuration
  - Deployed database  and Dataflow VMs in the same region for latency optimization.
  - Optimized database statement and idle timeouts and Fine-tuned network and JDBC socket timeouts.
- Layer 3: Database Configuration
  - Connection pooling (HikariCP).
  - Optimized Table structure
  - Strategic use of PostgreSQL features (ON CONFLICT, Unlogged Tables).



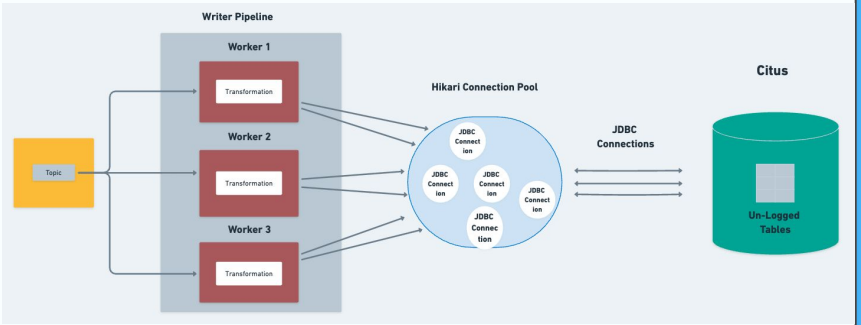
# Design Principle – Write Stage

- Layer 1: Pipeline Configuration
  - Batched JDBC writes.
  - Buffering via Apache Beam windows for deduplication.
- **Layer 2: Network Configuration**
  - **Deployed database  and Dataflow VMs in the same region for latency optimization.**
  - **Optimized database statement and idle timeouts and Fine-tuned network and JDBC socket timeouts.**
- Layer 3: Database Configuration
  - Connection pooling (HikariCP).
  - Optimized Table structure
  - Strategic use of PostgreSQL features (ON CONFLICT, Unlogged Tables).



# Design Principle – Write Stage

- Layer 1: Pipeline Configuration
  - Batched JDBC writes.
  - Buffering via Apache Beam windows for deduplication.
- Layer 2: Network Configuration
  - Deployed database  and Dataflow VMs in the same region for latency optimization.
  - Optimized database statement and idle timeouts and Fine-tuned network and JDBC socket timeouts.
- **Layer 3: Database Configuration**
  - **Connection pooling (HikariCP).**
  - **Optimized Table structure**
  - **Strategic use of PostgreSQL features (ON CONFLICT, Unlogged Tables).**

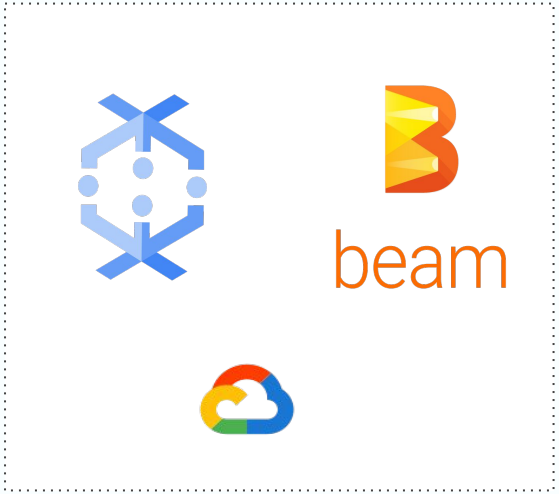


# Multi-cloud stream processing with Apache Beam

BEAM SUMMIT NYC 2025

# Apache Beam Superpower – Multi-cloud & Engine Flexibility

- Beam's unified model facilitates batch/stream unification.
- Multi-cloud: Operate seamlessly between GCP and AWS.
- Minimal code changes needed to switch from GCP Dataflow to Apache Flink.




# TRM Labs is Hiring

## Join the fight against

- **Fraud and Scams**
  Supporting victims of investment fraud with Massachusetts Attorney General
- **Fentanyl Crisis**
  Disrupting a prominent fentanyl vendor with Homeland Security Investigations
- **Terrorist Fighting**
  Tracking ISIS use of cryptocurrency
- **Cybercrime**
  Taking Down Qakbot malware with the FBI
- **Hacks**
  Seizing $12 billion of stolen Bitcoin


www.trmlabs.com/careers

Vijay Shekhawat

# QUESTIONS?

BEAM SUMMIT NYC 2025

vijay@trmlabs.com
linkedin.com/in/thevijayshekhawat/