IMS CDC to Kafka Performance and Tuning

Scott Quillicy
SQData Corporation

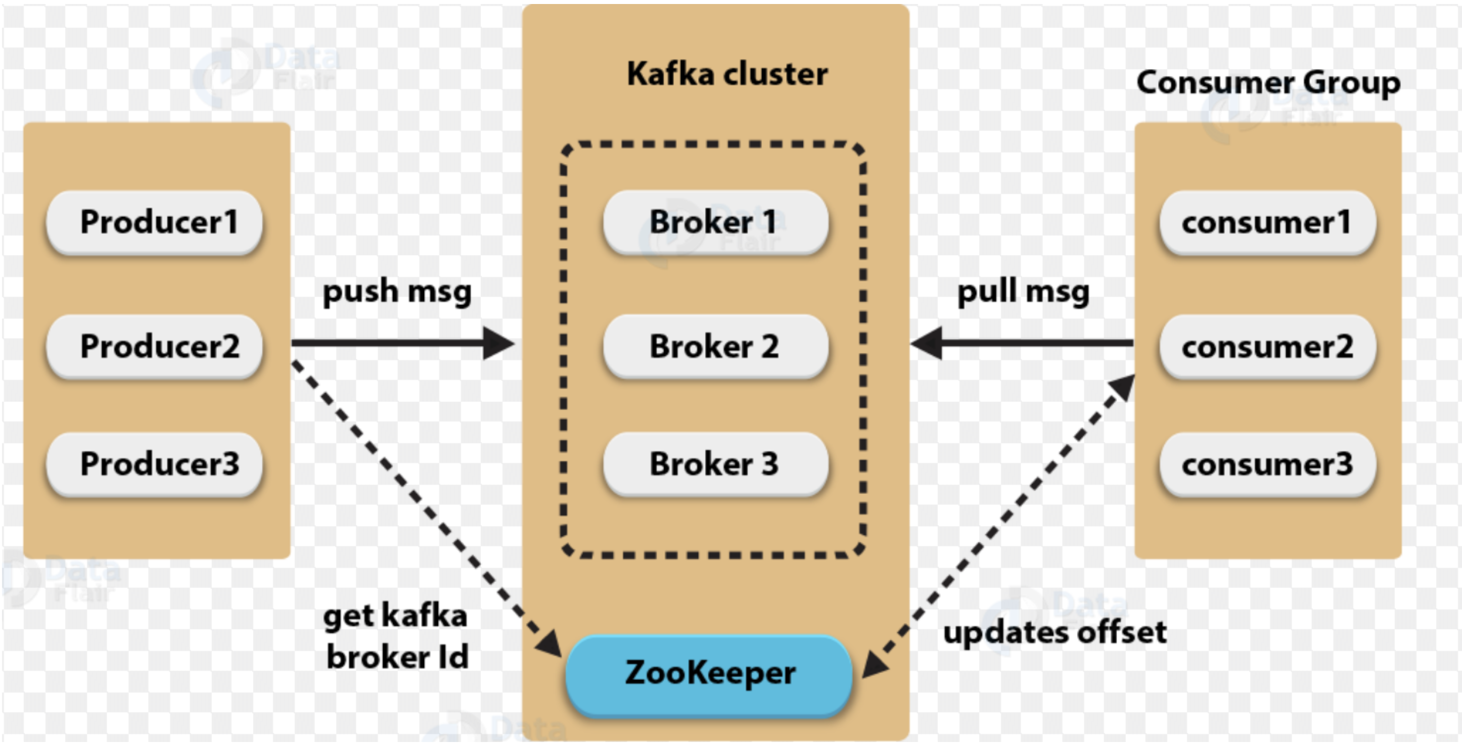
09-April-2019

Agenda

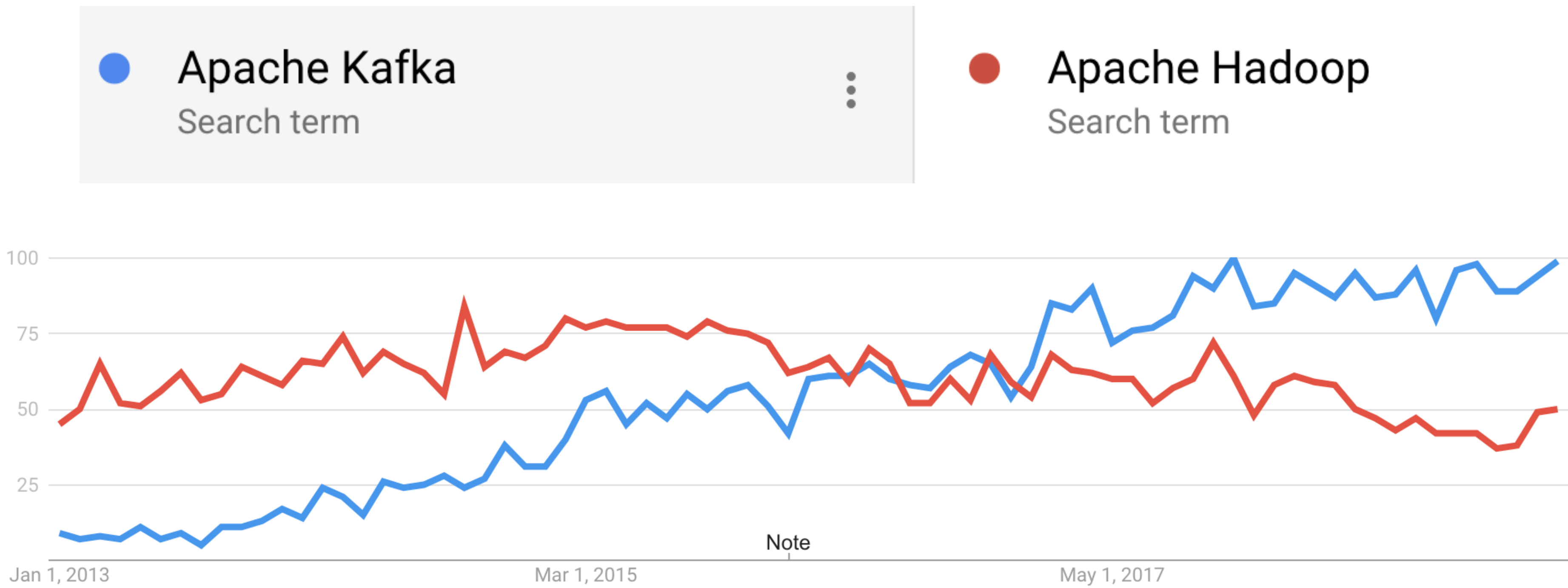
- Introduction
- Overview of Apache Kafka
- Key Performance Considerations for CDC Streaming to Kafka
- Initial Load Configuration and Performance
- IMS CDC Streaming Configuration and Performance
- Kafka Producer Performance Factors
- Common Operational Issues-→ What to do When Things Go Wrong
- Q&A

What is Kafka?

- Ultra-Fast Distributed Stream Processing Platform for Big Data
- Open Source – Very Large Community
- Publisher / Subscribe Messaging System
- Highly Scalable, Durable and Fault-Tolerant
- Topics (queues) and Topic Partitions are Separate Log Files



Interest Over Time → Kafka & Hadoop



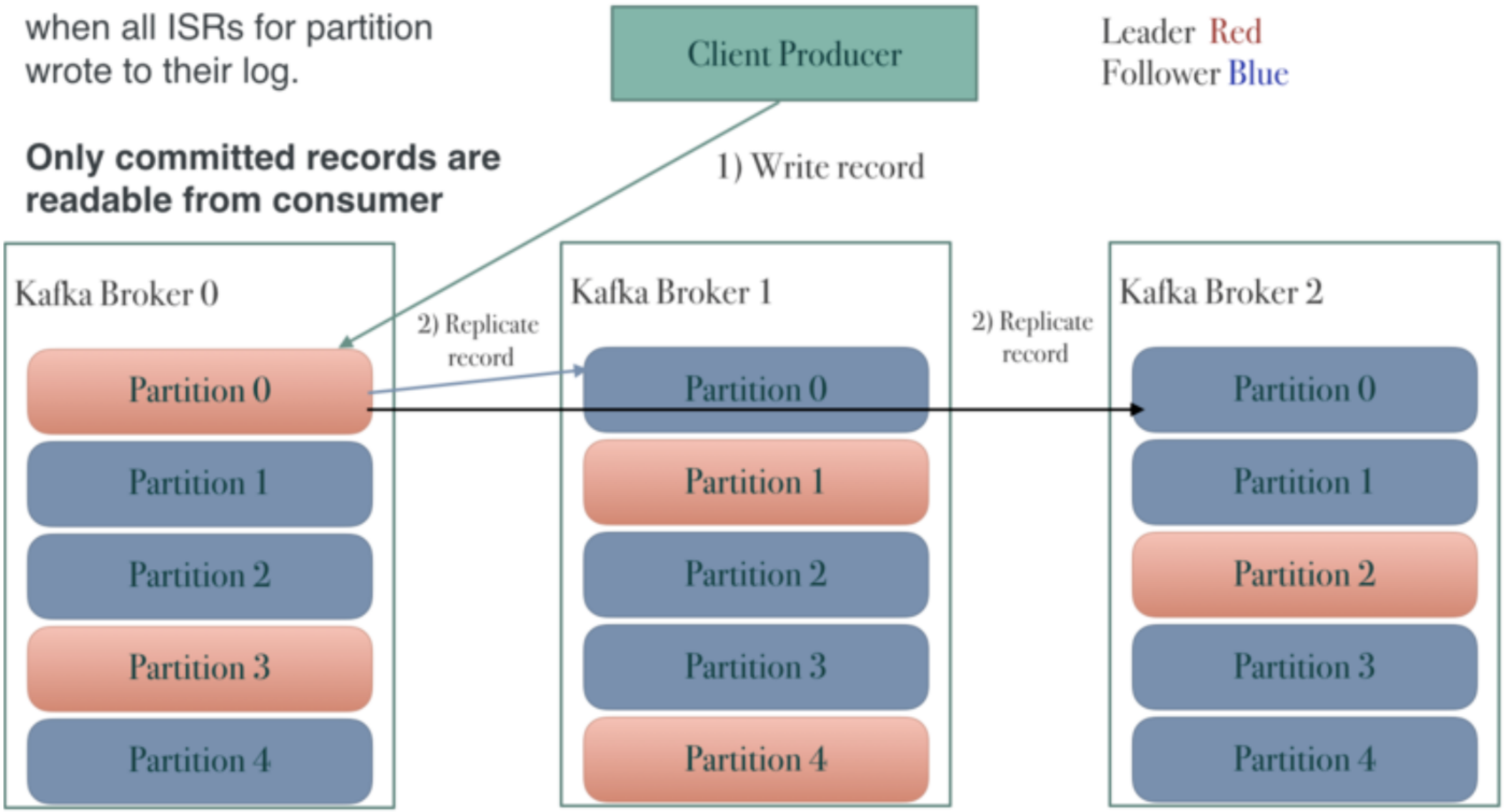
Source: Google Trends

Factoring in The Great Divide

Replication within Kafka

Record is considered "committed" when all ISRs for partition wrote to their log.

Only committed records are readable from consumer



Source: dzone.com

IMS Transactions & Kafka's Commit Scope

- Interesting Mix of ACID and BASE
- Source IMS is ACID → Commits and Rollbacks
- Target Kafka is BASE** → No Commits / Rollbacks

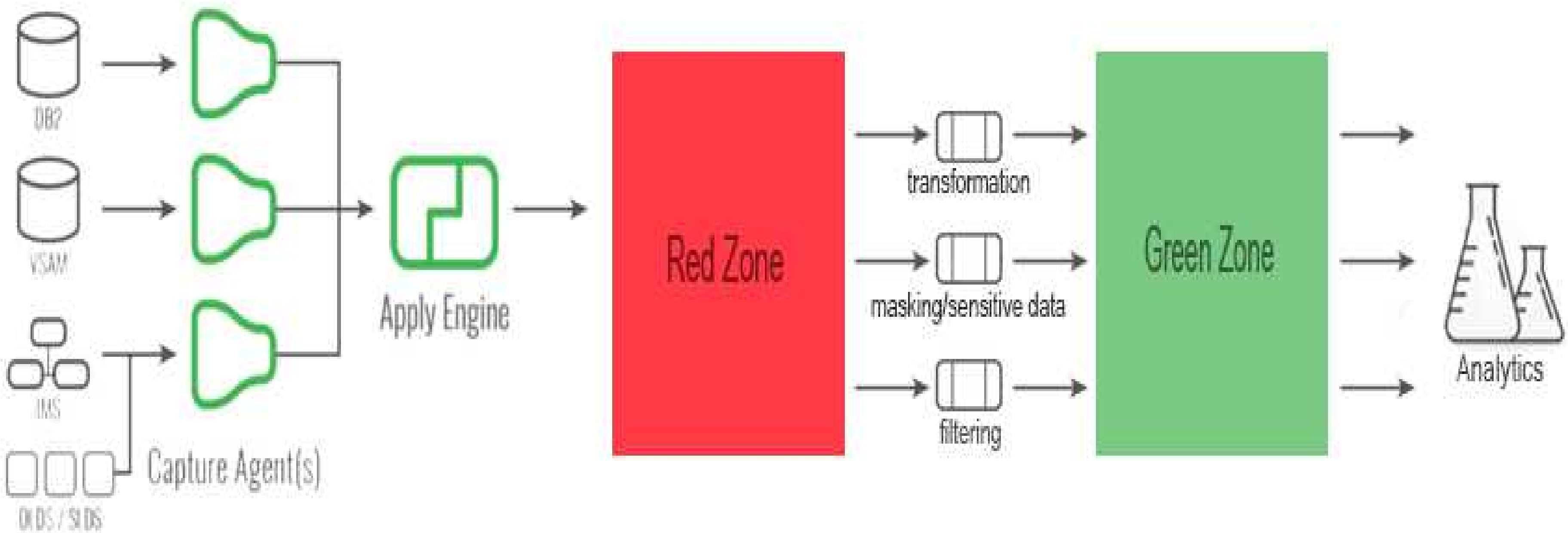
Kafka Commit Scope:

- ✓ Each Message is Acknowledged
- ✓ Not Transaction Based** → Each Topic is Acknowledged Independently
- ✓ **At Most Once** → Fastest - Could Lose Data
- ✓ **At Least Once** → Durable - Could have Duplicates
- ✓ **Exactly Once**
  - Msgs are Logged Exactly Once – No Duplicates
  - Slightly Additional Overhead on Producer

- What Does this Mean to Us?
  - ✓ Commit Scope based on Business Requirements
  - ✓ Suggest Planning for Duplicate Messages

** Evolving Quickly to Transaction Based Support

Common CDC Streaming Deployment Model



Capture & Ingestion

Initial
Data Landing
(Highly Secure)

Sanitized Data Landing
(Ready for Consumption)

Key Performance Aspects

- Throughput / Latency Primarily Depends on Speed of the Target
- Fortunately, Kafka is a Very Fast Target

Top Items Affecting Performance / Throughput

- Message Size
- Component Scaling Factor
- IMS Initial Load Configuration – Data Volume
- IMS CDC Streaming Configuration – Trx Size and Arrival Rate
- Kafka Configuration

Message Size

- Small Messages Perform Better than Larger Ones (a bit obvious)
- It Boils Down to the Number of Bytes per Message
- IMS Segments can be Large
- Updates can Contain Before and After Images

So...What to Do?

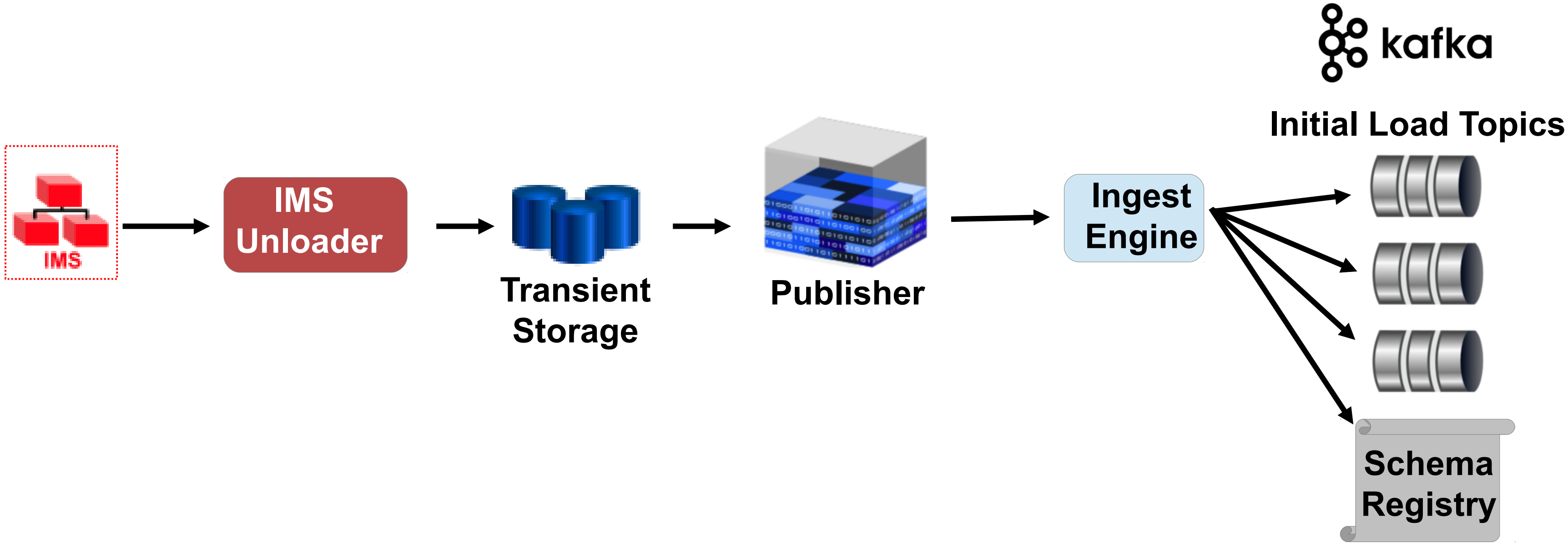
- Use Avro as the Target Data Format
  - ✓ A Condensed Version of JSON
  - ✓ JSON Typically Used for Data Validation Only → Can be Easily Read
  - ✓ Avro Messages Roughly the Size of Source Segment (x 2 for Updates)
- Reduce the Number of Fields in the CDC Message
- Use Compression
  - ✓ Reduces Footprint on Disk and on the Wire
  - ✓ lz4 Appears to be Most Stable Method

Sample IMS CDC Record in JSON Format


```
{"object_name":"IMSDB01.SEG02",
   "alias":"SEG02",
   "stck":"d4c4b51993db0000",
   "timestamp":"2018-08-12T11:11:18Z",
   "change_op":"U",
   "seq":"2",
   "parent_key":{
      "seg1.key1":12345
   },
   "after_image":{
      "fname":"MARY",
      "lname":"JOHNSON",
      "city":"CHICAGO",
      "amount":"4087.66"
   },
   "before_image":{
      "fname":"MARY",
      "lname":"JOHNSON",
      "city":"CHICAGO",
      "amount":"2964.32"
   }
}
```

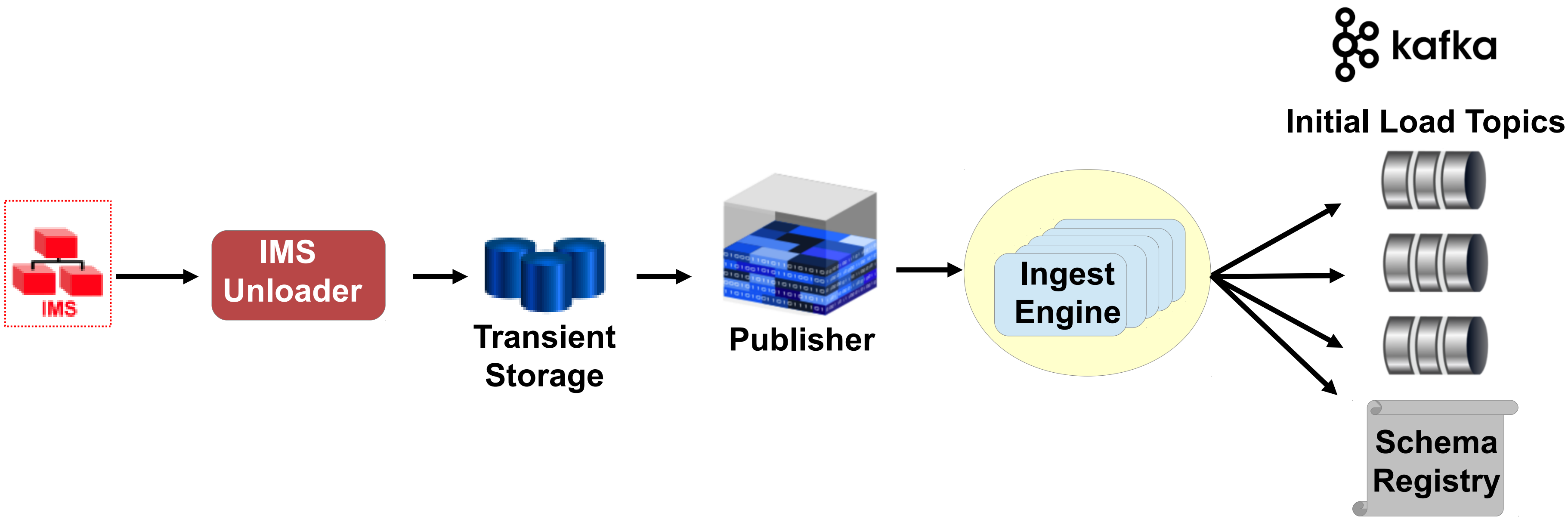
The Initial Load

- ✓ Required to Prime the Target(s)
- ✓ Load via Mass Inserts → There are No 'Traditional' Utilities on the Target Side
- ✓ Alternative: Online Snapshots → FTP → Transform → Ingest into Kafka
- ✓ Need to be Able to Scale on Both the Source and the Target Sides
- ✓ **Important:** Need to be Able to Run Unloads Against Live Databases
- ✓ **Recommend** → Use a Separate Set of Topics for the Initial Load



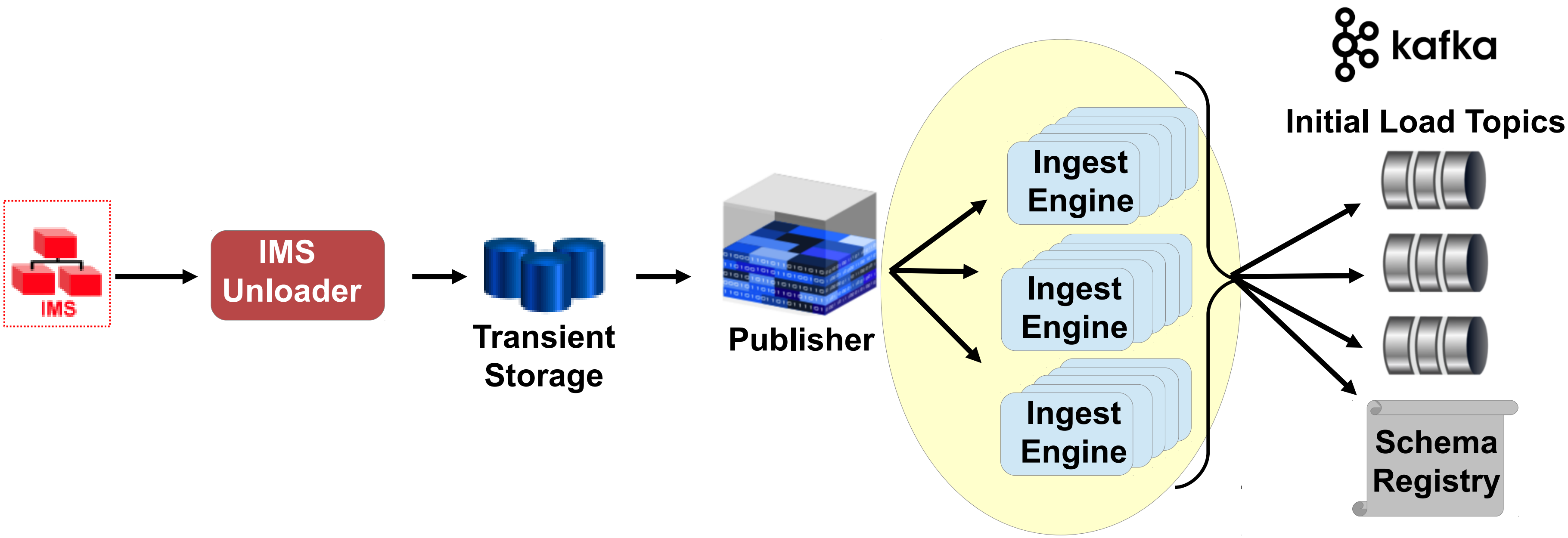
The Initial Load → Target Side Scaling

- ✓ Parallelize the Ingest Engine
- ✓ One (1) Publisher Subscription
- ✓ **Desired Behavior** → Publisher Latency Should be within a Tolerable Limit



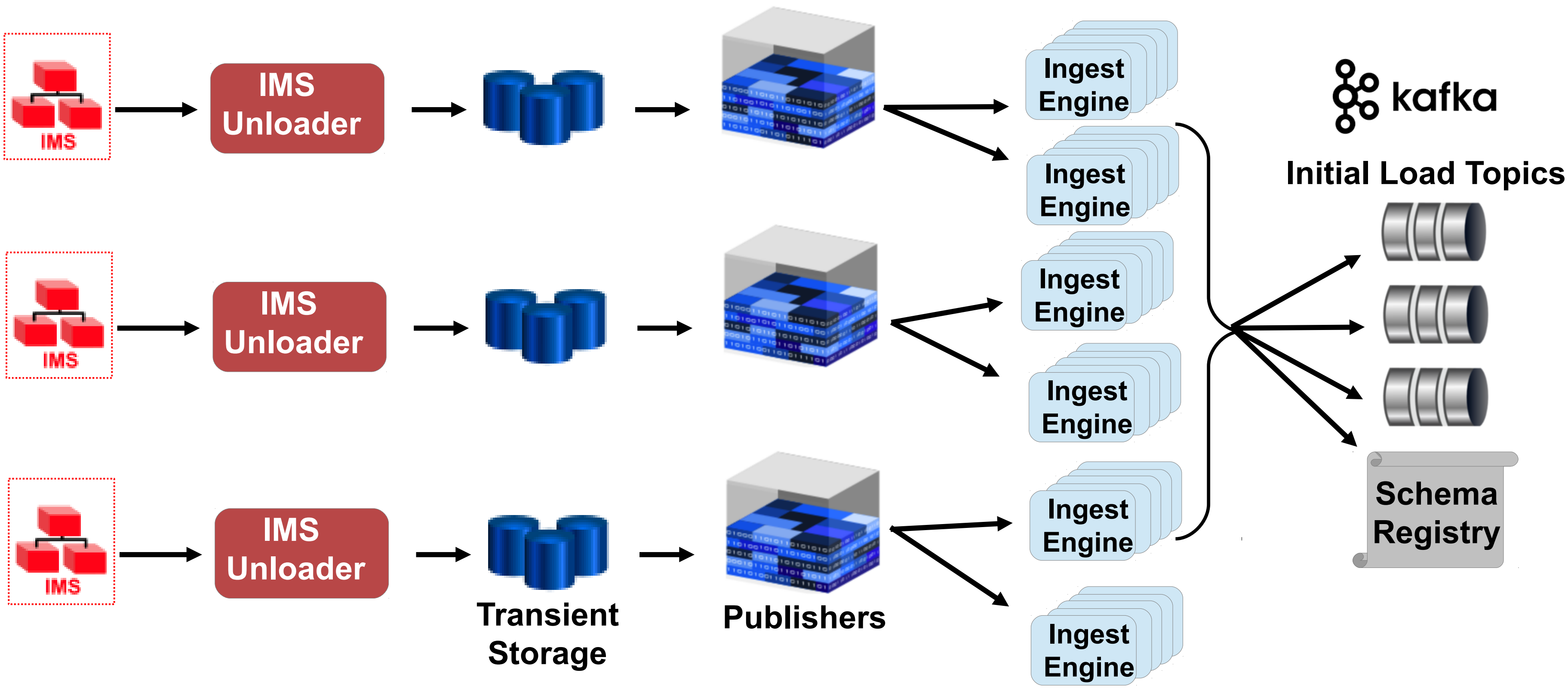
The Initial Load → Source Side Scaling

- ✓ Multiple Publisher Subscriptions → Split by Database / Partition
- ✓ Parallel Ingest Engines on Target Side – 1 per Subscription
- ✓ **Desired Behavior** → Publisher Latency Should be within a Tolerable Limit



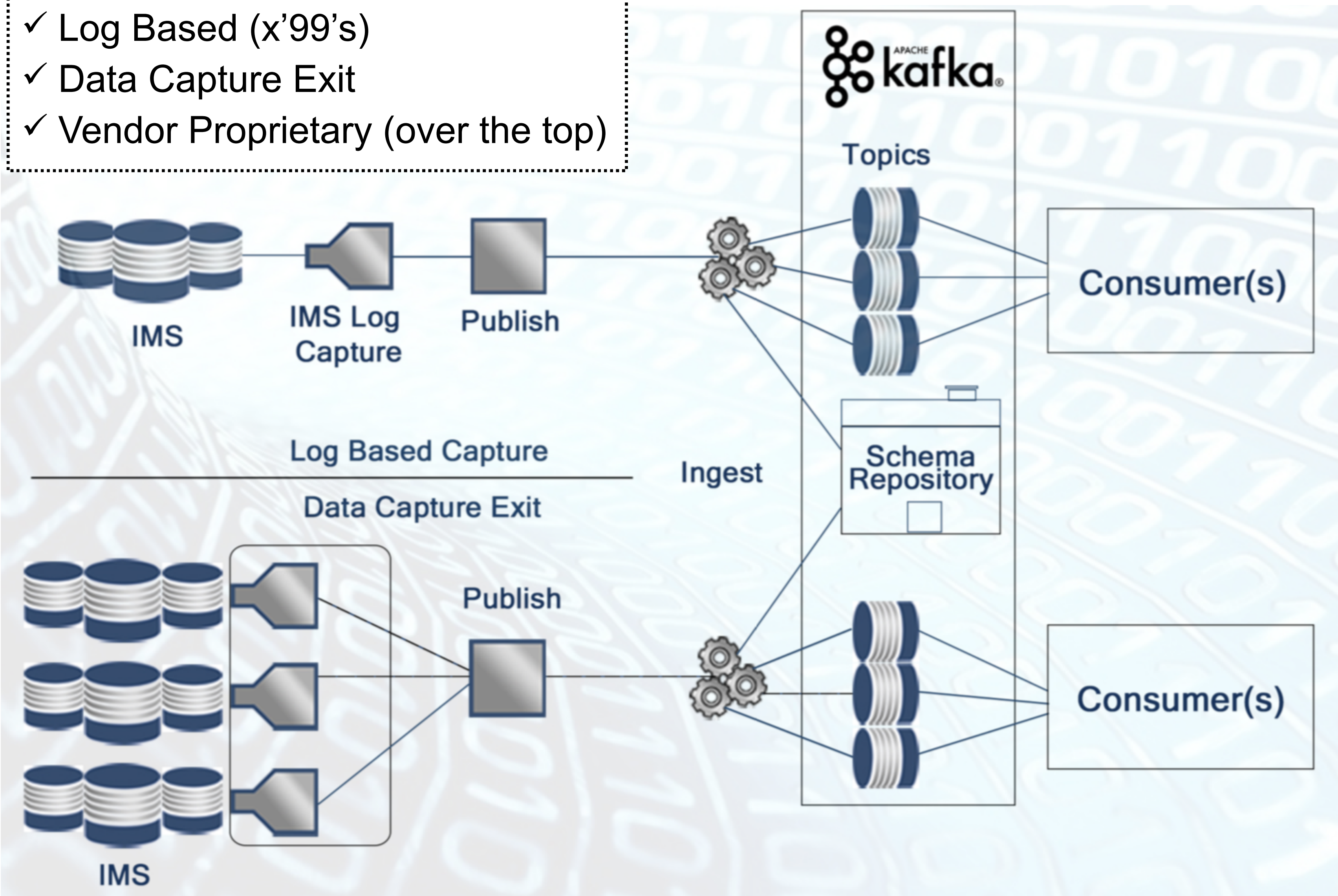
The Initial Load → Source Side Scaling...

- ✓ Multiple Unloaders and Publishers
- ✓ Used for High Data Volume
- ✓ **Desired Behavior** → Minimal Time for the Initial Loads



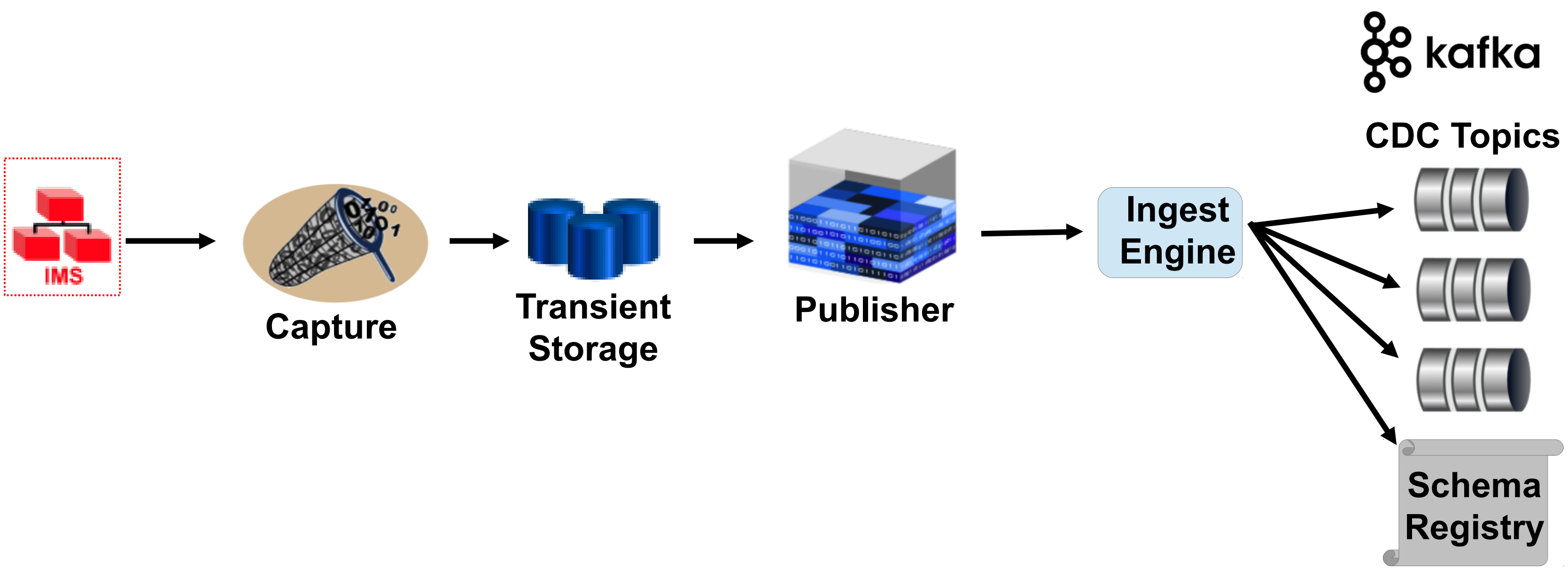
IMS CDC Streams → Methods of Capture

- ✓ Log Based (x'99's)
- ✓ Data Capture Exit
- ✓ Vendor Proprietary (over the top)



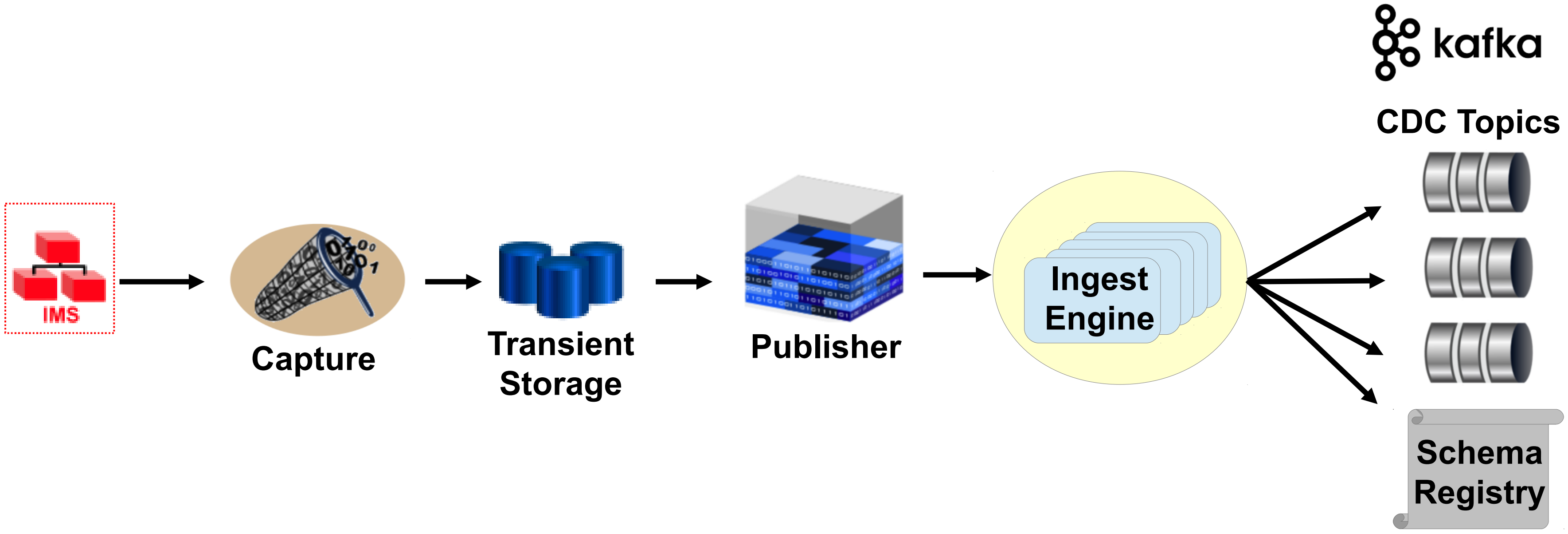
IMS CDC Streams

- ✓ Recommended - a Similar Architecture to Initial Loads
- ✓ Start with Basic Configuration and Scale Up as Required
- ✓ Need to be Able to Scale on Both the Source and the Target Sides
- ✓ **Recommend** → Use a Separate Set of Topics for the CDC Streams



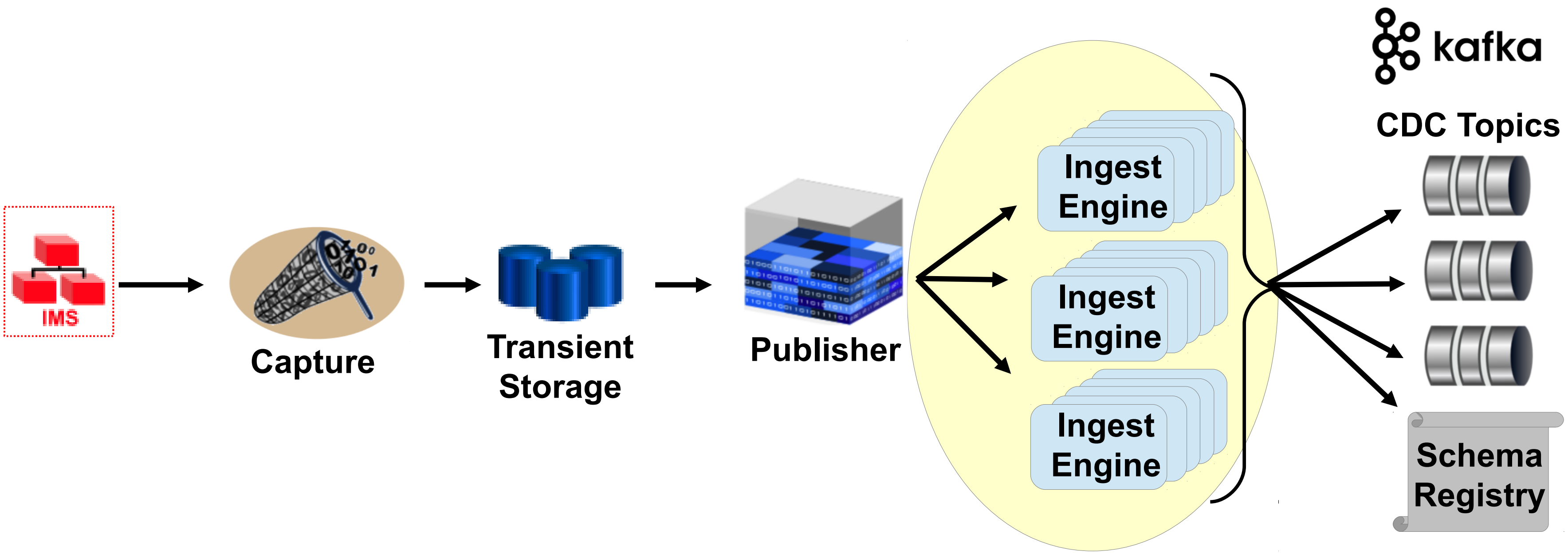
IMS CDC Streams → Target Side Scaling

- ✓ **Start Here First**
- ✓ Parallelize the Ingest Engine
- ✓ One (1) Publisher Subscription
- ✓ Order *May* Matter – Ingest Engines Need to be able to Handle Transaction Order
- ✓ **Desired Behavior** → Publisher Latency Should be within a Tolerable Limit



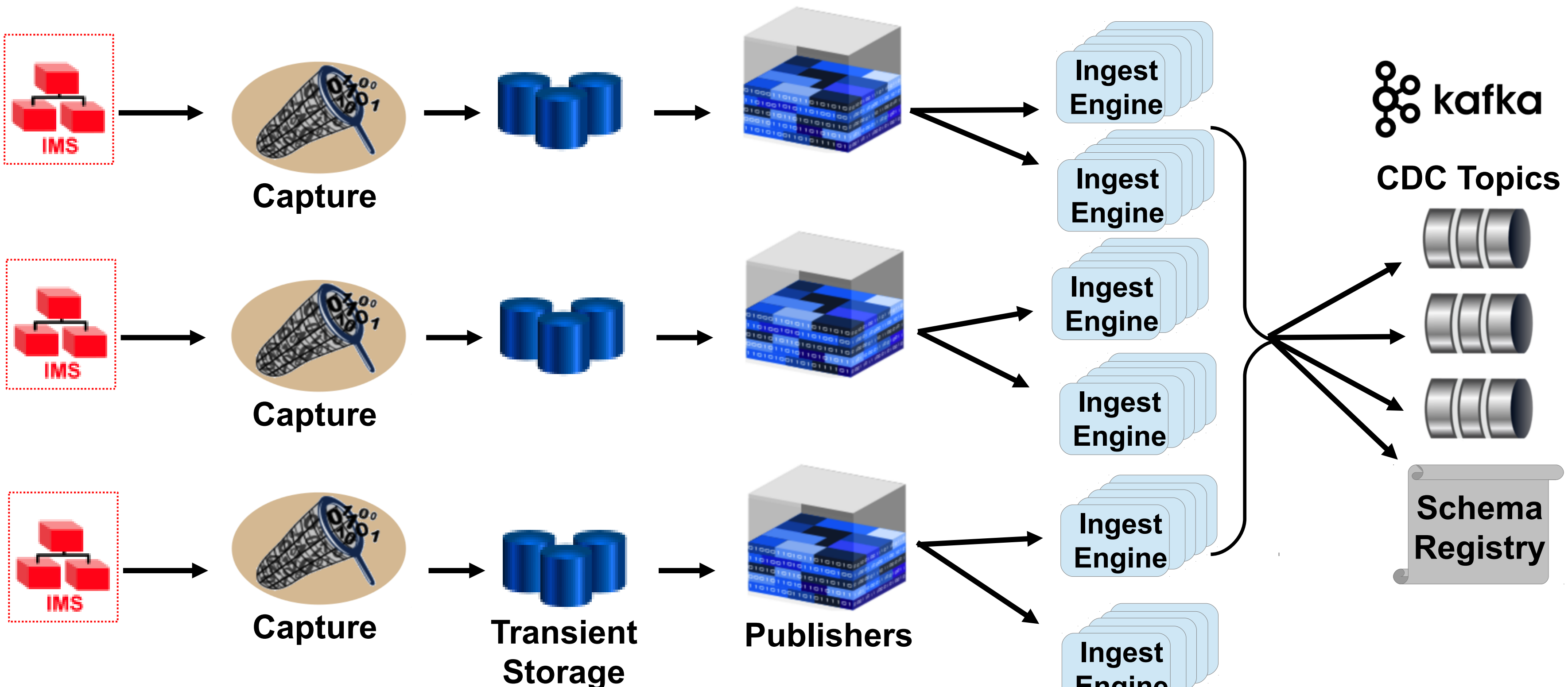
IMS CDC Streams → Source Side Scaling

- ✓ Multiple Publisher Subscriptions → Split by Database / Partition
- ✓ Parallel Ingest Engines on Target Side – 1 per Subscription
- ✓ **Desired Behavior** → Publisher Latency Should be within a Tolerable Limit



IMS CDC Streams → Source Side Scaling...

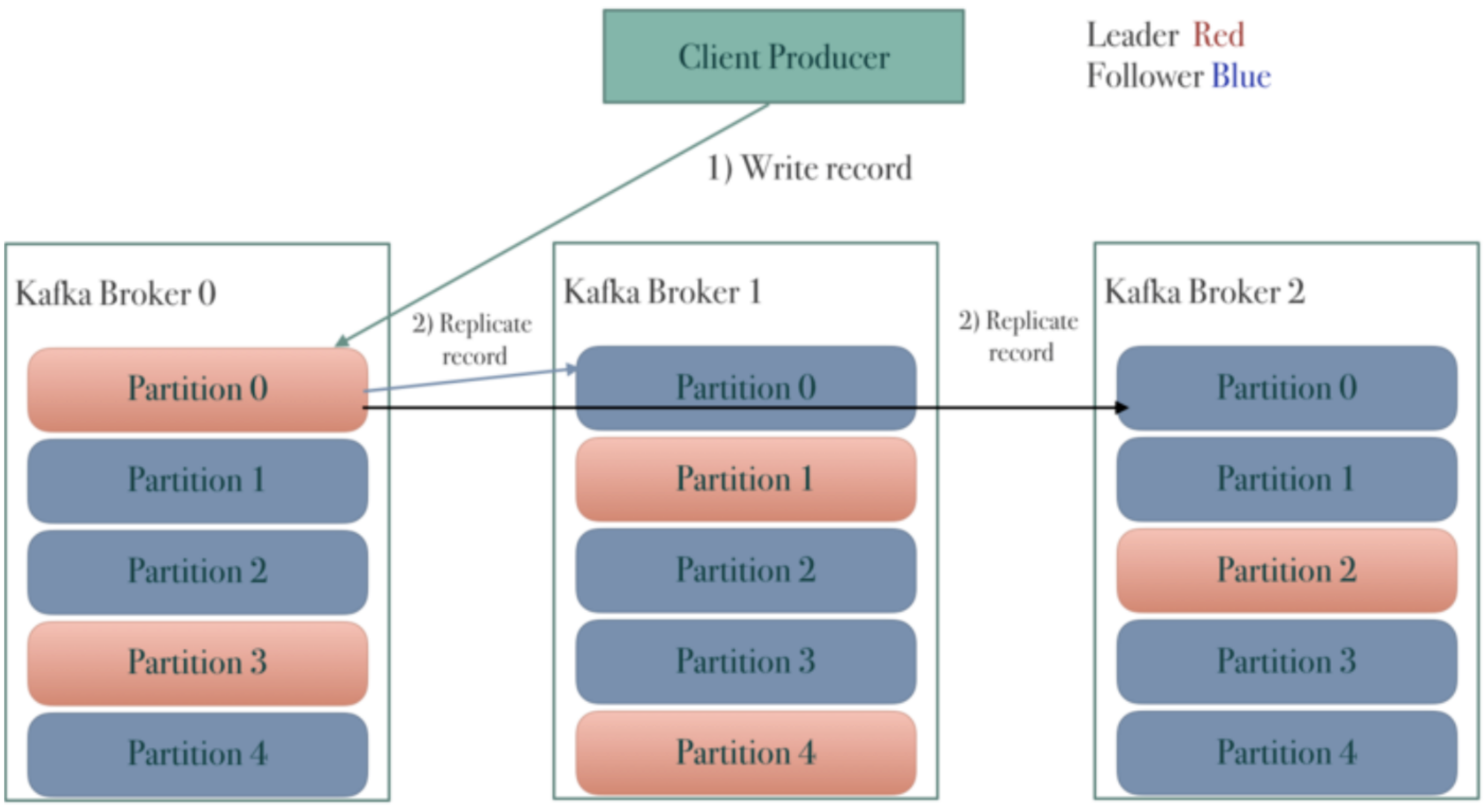
- ✓ Multiple Captures / Publishers → Split by IMS Subsystem / Data Sharing Partner
- ✓ Applicable for Segregated Workload (Online vs Batch)
- ✓ Suggest Combining Online SSIDs and Splitting Out Batch SSIDs



Kafka Replication Factor and Throughput

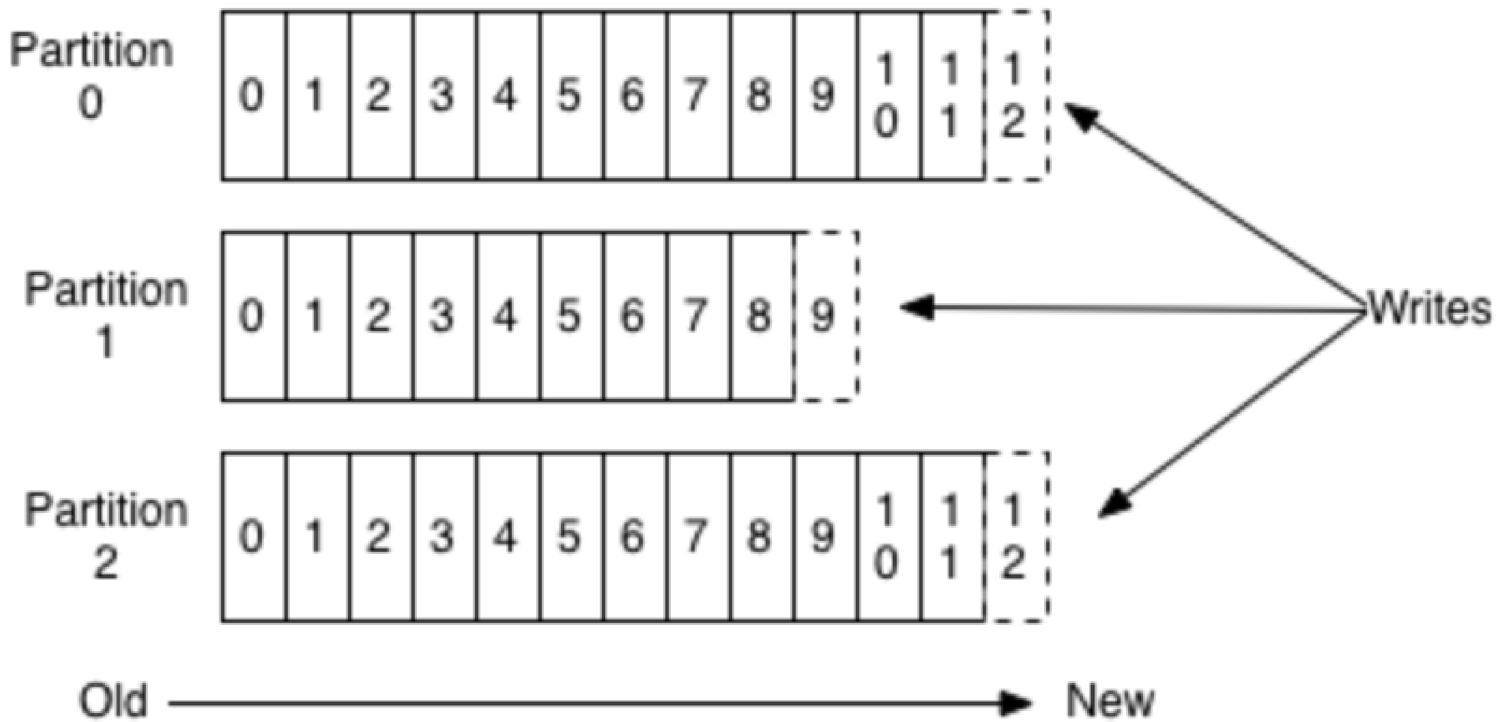
Acknowledgements – Receipt of a Message

- ✓ **Acks=0** → Fire & Forget → No Delivery Guarantee → Fast but Less Reliable
- ✓ **Acks=1** → Ack after Leader Receives Message → Highly Durable but Not Max
- ✓ **Acks=all** → Ack after All Replicas Receive Message → Slowest but Most Durable



Batch Size Affect on Throughput & Latency

- ✓ Topic are Log Files – Offsets are RBAs – Writes Append to the Tail
- ✓ Kafka Likes Batches → Can Write Multiple Messages without Repositioning
- ✓ Small Batches → Low Latency - Ideal for Near Real Time → Recommended
- ✓ Large Batches → Increase Latency and Throughput



Kafka Topic Partitions

The Upside

- Increases Throughput via Parallelism – Producer and Consumer
- Highly Recommended

The Downside

- Increases Chances of Unavailability if a Broker Abends
  - ✓ It Takes Longer to Assign a New Leader if Many Partitions
  - ✓ Est 5ms per Topic Partition, so 10,000 Partitions will Take about 50 secs
- May Increase End-to-End Latency
  - ✓ Consumer can only See Messages that have been Fully Replicated
  - ✓ Est about 20ms to Replicate 1,000 Partitions between Brokers
  - ✓ Somewhat Alleviated if Running a Large Cluster

Recommendation

- 100 x b x r, where *b* is the Number of Brokers and *r* is the Replica Factor

Common Operational Situations

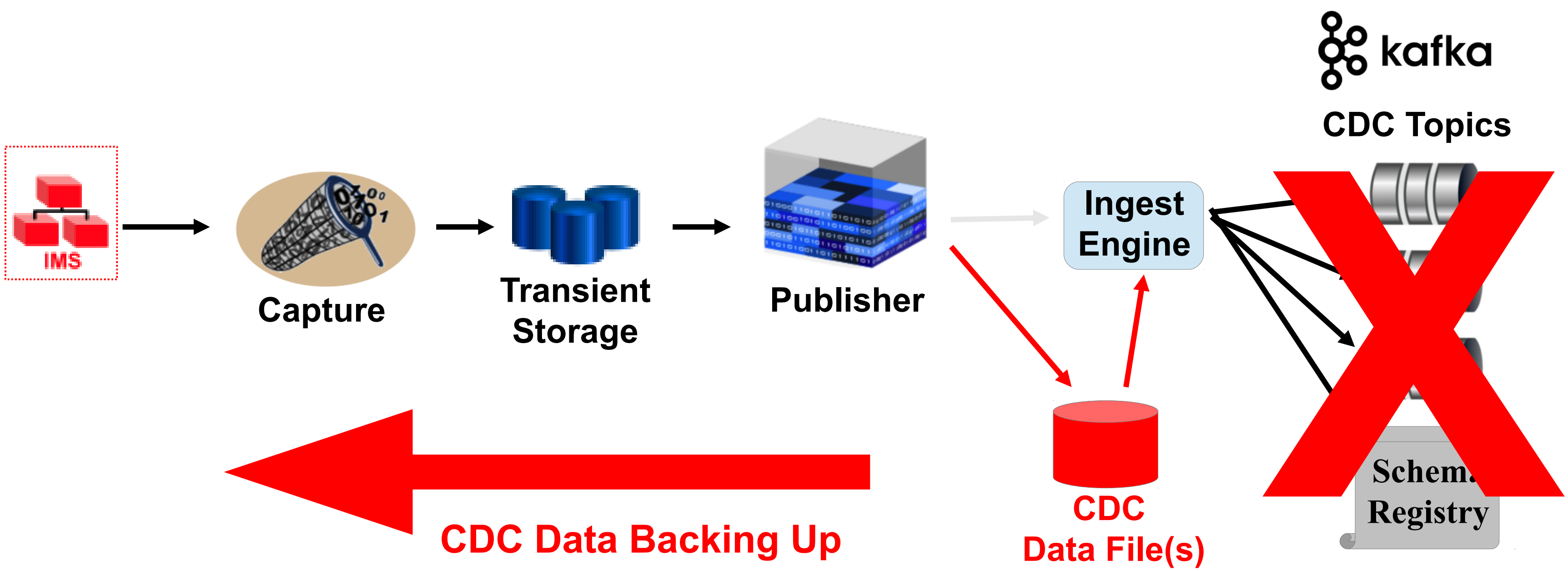
- **Target Side**
  - ✓ Kafka Cluster Outage
  - ✓ Kafka Slow

- **Network Drop**
  - ✓ Similar to Target Outage, but with a Much Shorter Duration
  - ✓ Tool Should be Able to Restart Where it Left Off → No Data Loss

- **Source Side**
  - ✓ Compressed PRILOG in IMS RECON
  - ✓ Capture / Publisher Outage
  - ✓ Capture / Publisher Slow

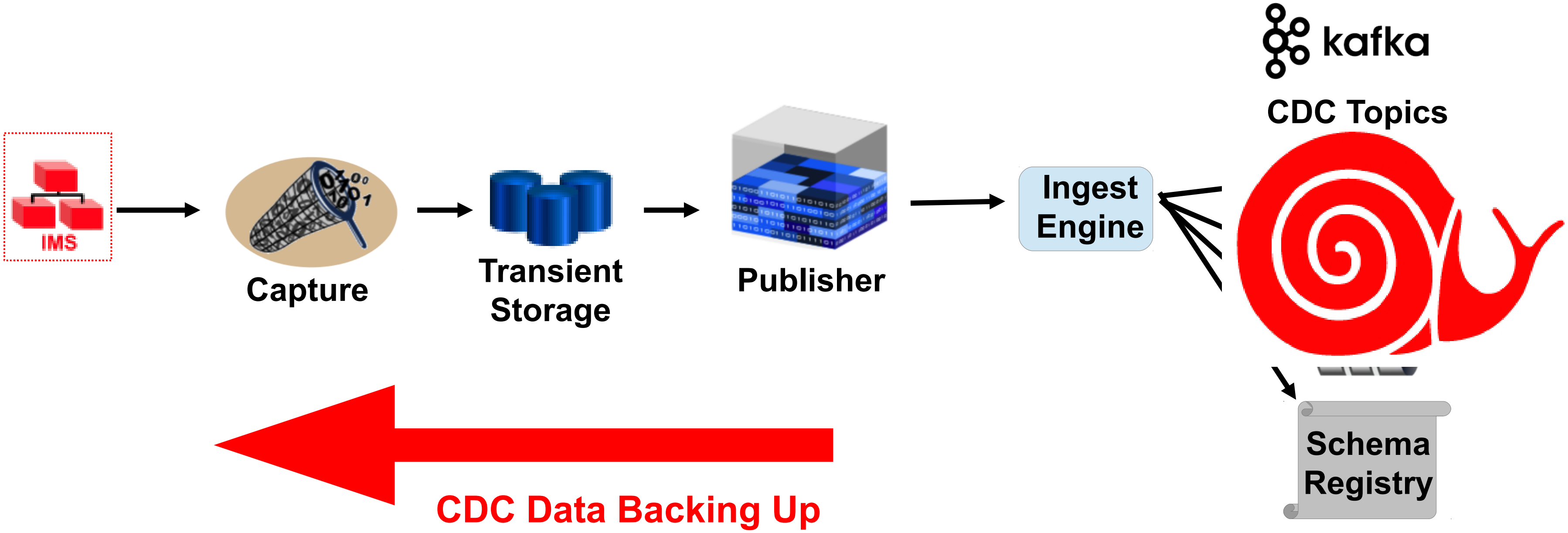
Kafka Cluster Outage

- ✓ Target Unavailable to Receive Data → Ingest Engine(s) will Fail
- ✓ Rare for Kafka to Lose the Whole Cluster, but it Can Happen
- ✓ Data will Start Backing Up on the Source Side
- ✓ **Option 1:** Wait Until Cluster is Available and Restart
  - OK if for a Short Period of Time – Depends on Source Transaction Rate
  - Eventually, You Will Reach the Point of No Return
- ✓ **Option 2:** Spin Off the CDC Data to a File on the Server – Process When Cluster Comes Back Up



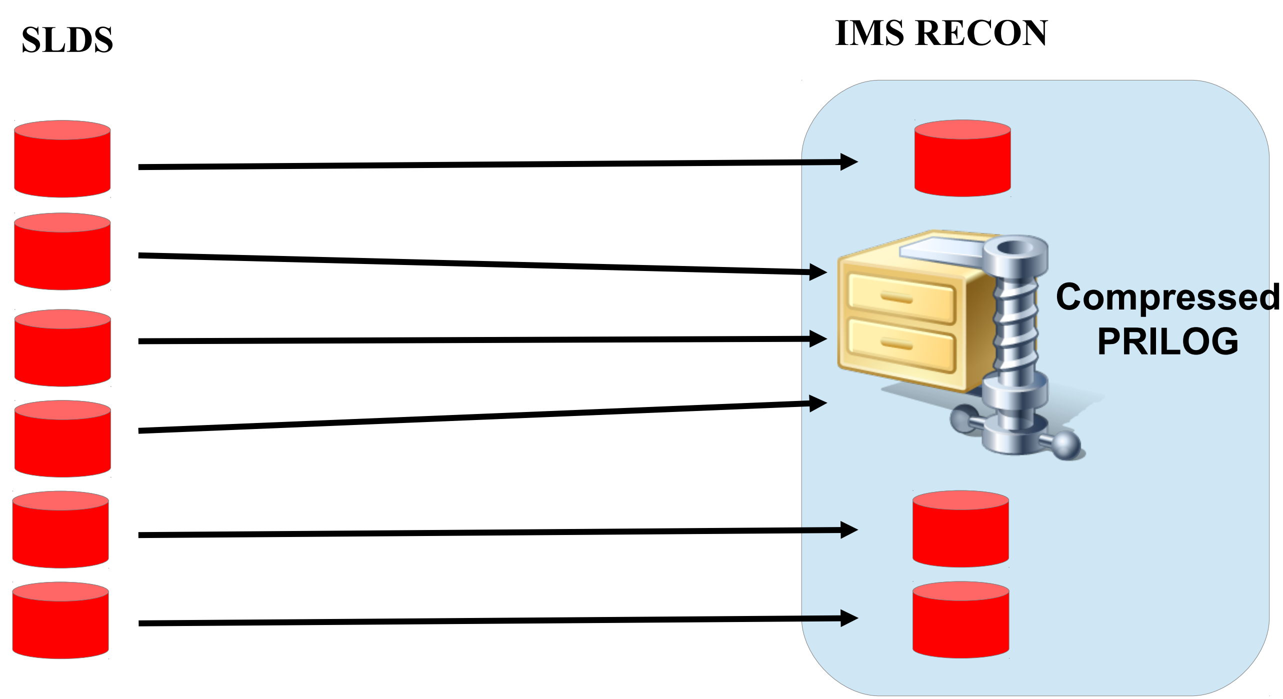
Kafka Cluster Slow

- ✓ Throughput Measured with a Calendar – Data Backs Up on the Source Side
- ✓ **Common Causes**
  - One (1) or More Brokers Down – Correct and Restart
  - Out of Memory: Set queued.max.requests at a Reasonable Value (more can be trouble)
  - Memory Buffers: Should be Entirely in RAM
  - Log Files: Review Strategy and Settings – There are Many
- ✓ Important: Leverage Monitor Tools such as the Confluent Control Center
- ✓ May Need to Back Off CDC Until Tuning has been Optimized



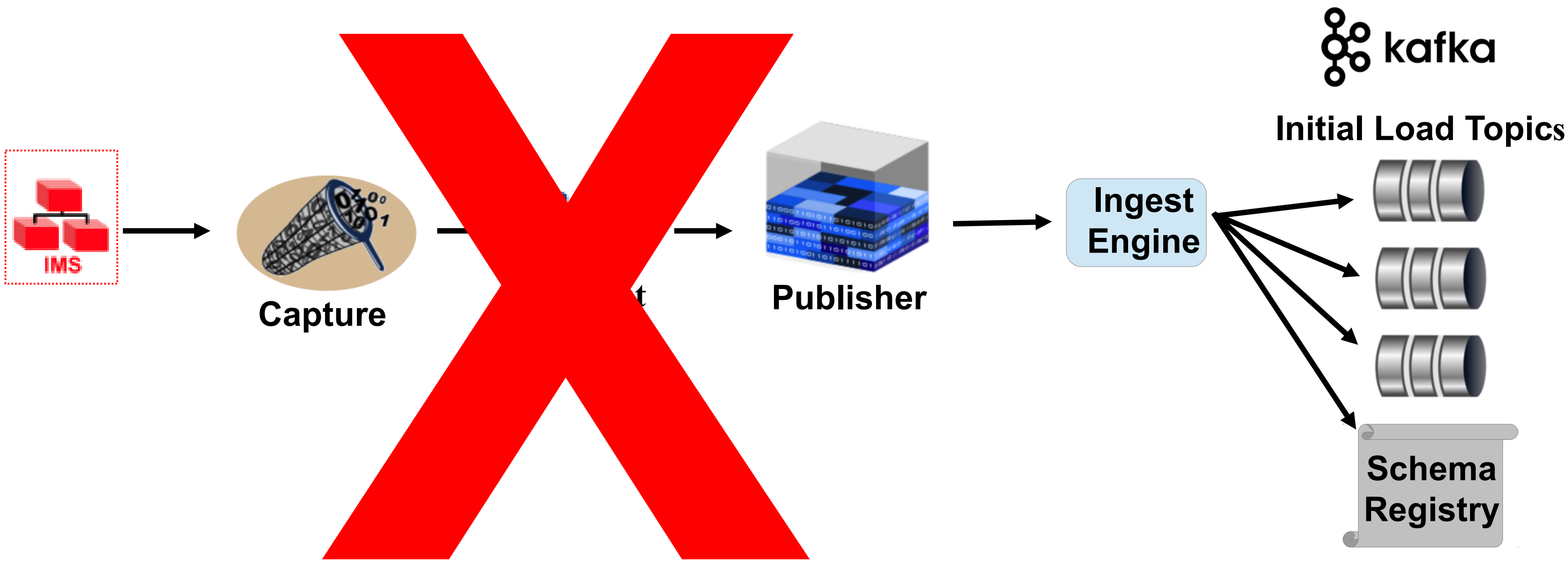
Compressed PRILOG in RECON

- ✓ SLDS Datasets Exist, but Entries Disappear from the RECON
- ✓ IMS Considers SLDS to be Inactive → CDC Thinks Otherwise
- ✓ SLDS May Contain CDC Data
- ✓ Can be Avoided by Increasing the DBRC Log Retention Period
- ✓ Replication Tool Should Provide a Method of Recovery
- ✓ If Tool Cannot Handle → Unwelcome Manual Intervention and Perhaps a Reload



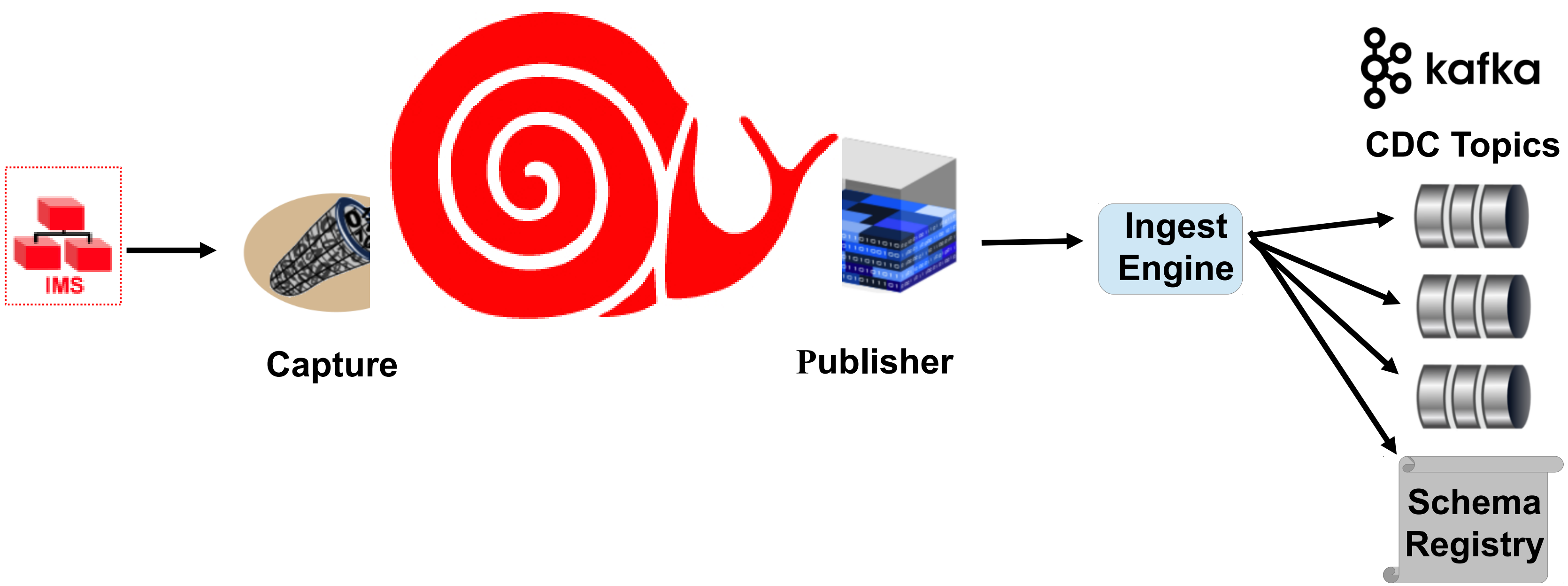
IMS Capture / Publisher Outage

- ✓ No Data Flowing – Downstream Folks Getting Concerned
- ✓ LPAR Down – Failover to another LPAR – Should Resume Where it Left Off
- ✓ Transient Storage Issue – Allocate More Space, if Possible
- ✓ May Take Awhile to Catch-Up
- ✓ Contact the Vendor ASAP if You Cannot Restart



IMS Capture / Publisher Slow

- ✓ Kafka Not Doing Much → Issue is on the Source Side
- ✓ **Common Causes**
  - Busy System - Processes are Running at Low Priority – Increase Dispatching Priority
  - Abnormally Large Units-of-Work – Exclude ‘Purgers’ (mass deletes)
  - Network TCP/IP Window Size Too Small – Adjust to Avoid Auto-Adjusting
- ✓ **Important:** Make Sure CDC Tool Provides Appropriate Monitoring Metrics



Summary

- **Keep Ingestion Method as Simple and Consistent as Possible**
  - ✓ Initial Loads
  - ✓ CDC Streams
- **Understand the Business Requirements**
  - ✓ End-to-End Latency Expectations
  - ✓ Near Real Time may Require More Scaling
- **Scale to Satisfy Business Requirements → Don't Overdo It**
  - ✓ Dependent Upon Transaction Volume
  - ✓ Low Latency may Require More Scaling
- **Have a Solid Recovery Plan**
  - ✓ Spinning Off CDC Records to Disk on Target Side
  - ✓ Reloading Source Data
- **Be Patient with 'The Great Divide' → You Will Need Every Bit**
- **Select the Right Tools for the Job**


Q&A

IMS CDC to Kafka Performance and Tuning

Scott Quillicy
SQData Corporation

09-April-2019