

Data Provenance and Lineage Tracking in Big Data Pipelines.

D Pavan Kumar, Mrs Shruthi M T

Student, Department of MCA, BIET, Davanagere, India

Assistant Professor, Department of MCA, BIET, Davanagere, India

ABSTRACT

In the information technology age of today Big Data processing pipelines have evolved from simple extract-and-load routines into complex multilevel constructs comprising various data assets complex transformations and machine learning deployments. Since such processing patterns become increasingly entwined with critical business decision-making the "black box" nature has made Big Data processing a critical bottleneck. The aspect of data traceability regarding its origin transformation points and integrity of its final output has turned into a huge challenge confronting all stakeholders. This condition has come to be referred to as the problem of data lineage which destroys trust in data slows debugging processes and makes HIPAA or CCPA impracticable. This project **Data Provenance & Lineage Tracking in Big Data Pipelines** proposes a comprehensive framework designed to capture store and visualize the lifecycle of data as it flows through modern processing architectures. By implementing non-intrusive instrumentation agents within pipeline orchestration tools (such as Apache Airflow or Spark) the system automatically intercepts metadata events at every stage of the data lifecycle—creation modification aggregation and deletion. Are synthesized into a directed acyclic graph (DAG) model and stored in a high-performance graph database enabling real-time querying of data relationships.

Keywords: Granular Lineage Visualization, Interactive Visual Impact Analysis Graph, Version Control for Data Pipelines, Data Quality & Health Integration

1. INTRODUCTION

The "Data Provenance & Lineage Tracking in Big Data Pipelines" project is an enterprise-grade software solution developed to address the critical visibility gaps inherent in distributed data processing

environments. In modern data engineering a single dataset may traverse through dozens of systems—starting from IoT sensors or transactional databases moving through message queues like Kafka being processed by distributed engines like Apache Spark or Flink and finally resting in data warehouses or feeding into predictive machine learning models. In this complex journey the ability to answer the question "Where did this number come from?" becomes exponentially difficult.

II. LITERATURE REVIEW

Lately, Big Data researchers have been pretty vocal about how important it is to track where data comes from and what happens to it—basically, who touched the data, and how, every step of the way. As data pipelines keep getting more tangled and complex, it's just not easy anymore to pinpoint the original source or follow the whole path data takes through your system. The old way—just logging everything in files—doesn't cut it. You end up with a scattered history that doesn't really show the full story, especially across distributed systems.

A lot of teams now turn to graph-based models for tracking lineage. Think of it like this: each dataset and process is a node, and every relationship—what feeds into what, or how one step affects the next—gets its own edge. With this setup, you can see all the dependencies at a glance, and when something goes wrong, it's a lot easier to dig in and find the cause. This approach also lets you do impact analysis without guesswork or spending hours reading logs. Plus, there's a big push for collecting this metadata automatically, instead of expecting people to document every change. Automated collection beats manual work for accuracy and scale—no surprise there.

There's another side to all this: compliance and audits. If you're dealing with regulations like GDPR

or HIPAA, you need rock-solid lineage tracking. The rules demand that you prove where your data came from and how it's been handled. Mess that up, and you're looking at trouble. So, after looking through the latest research, it's pretty clear: if you want real trust and transparency in your Big Data system, you need automated, scalable, and centralized lineage tracking. That's the only way to handle governance and make solid decisions you can stand behind.

III. EXISTING SYSTEM

In today's environment data lineage and provenance are tracked by a typical IT organization with a patchwork quilt of manual documentation ad hoc logging and tribal knowledge. Logging capabilities are provided by individual tools such as viewing job execution info on the Spark History Server searching Airflow logs to find scheduling errors or querying database system tables to find modification times.

This creates a patchwork system that lacks an integrated view of the data life cycle from the ingestion tiers (such as Kafka) the processing infrastructure (such as Databricks or Hadoop) all the way down to consumption interfaces (like Tableau or Power BI). During data problems the data engineer will have to navigate by hand from system to system possibly SSH-ing into each server and running grep over the large amount of unstructured logs to spot the correlations.

To add to the problem the documentation of the flow of data is often represented in static forms such as Microsoft Excel sheets Confluence pages or Visio diagrams. These are usually designed in the design phase by software developers who then maintain it manually. When software code changes evolve with hot-fixes being deployed in the production phase the documentation isn't generally kept updated. Eventually documentation drift affects software immensely sometimes to the point of the software design being completely different from what has been documented.

Limitations:

Fragmented and Siloed Visibility:

Data movement-related information is locked away inside a specific tool. "The database administrator knows how the database schema changes the data engineer knows how transformations are performed.

High Latency in Root Cause Analysis (RCA): The tracing of the root cause of the problem in a final report in a time- consuming task of the investigative kind. The engineers took days trying to ensure the problem was not in the transformation logic.

Inability to Support Regulatory Audits: Because of the rising regulatory requirement for data protection the auditors need assurance of the handling of the data. The handling of the current systems cannot guarantee the spread of the critical PII right.

Reactive rather than Proactive Debugging : The system is always reactive. Engineers only enter the lineage system when they have an issue. There are no methods for engineers to predict and show data on how well the pipe is doing before an issue manifests in other systems.

IV. PROPOSED SYSTEM

This new system is all about taking the headache out of manual documentation. Instead of scattered logs and endless paperwork, it's an automated platform that keeps track of metadata and its history for you. So, here's how it's set up: The team is going with a "Metadata Lake" architecture. Basically, it treats every metadata event as equal—no more cherry-picking what's important and what's not. Everything gets captured, making the whole tracking and management process way smoother and a lot less confusing.

"citizenship" rights as the data they describe. This design incorporates micro- services architecture which has three main levels: the Collection Layer Persistence Layer and Serving/Visualization Layer.

Talking about the Collection Layer the proposed system uses something called "probes" or "integrators." These are light libraries that can be conveniently appended at the end to Big Data platforms such as Spark Hive Kafka and Airflow. With every instance of accessing a particular data set there is an entity called probes that extracts the metadata namely schema number of records execution user and query logic and pushes a lineage event in JSON format to any Message Queue System.

The Persistence Layer uses (e.g. Neo4j or JanusGraph) as its main engine for persisting.is optimized for storing relationships. All information related to all data

sets jobs users is modeled as a "Node" while all operations (reads writes triggers) as an "Edge." This makes it very easy to identify non-linear relationships in a Big Data setup.

Advantages:

End-to-End Automated Visibility: You get a clear, unbroken look at your data's journey. No matter which technologies you're working with, you can track any piece of data every step of the way. There's nowhere for it to get lost or hidden.

Drastic Reduction in Debugging Time: Debugging goes way faster. With the system's visual tools, engineers spot errors right on the graph and see how different tasks depend on each other. What used to take days now happens in minutes. It's simple to find the exact job that's causing problems and fix it .failed and thus the list of the corrupted and stale data sets.

Increased Trust and Data Accuracy: The business stakeholders have the ability to trace their measures. The past is important in giving confidence that measures shown in the dashboard are derived from credible sources.

System Architecture

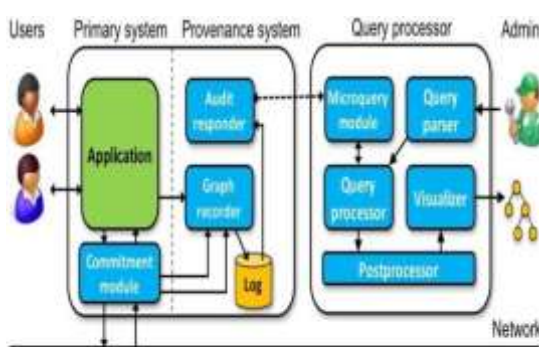


Figure 5: System architecture.

Fig 1. System Architecture

V. METHODOLOGY

The methodology of the Data Provenance and Lineage Tracking in Big Data Pipelines project follows a structured and systematic approach to ensure accurate tracking of data flow.

1. **Requirement Analysis** The first step involves

analyzing the Big Data environment to understand data sources, processing tools, and data consumers. The required metadata such as data origin, transformations, users, and timestamps are identified.

2. **Metadata Collection**

Automated listeners and probes are integrated with Big Data processing tools such as Apache Spark and Apache Airflow. These components capture metadata during data ingestion, processing, and storage without affecting pipeline performance.

3. **Metadata Storage**

The collected metadata is stored in a graph- based structure where datasets, jobs, and users are represented as nodes, and data flow relationships are represented as edges. This structure helps in managing complex dependencies efficiently.

4. **Lineage Graph Construction** The system links metadata events to build an end-to-end lineage graph. This graph shows how data moves from source to destination and how transformations are applied at each stage.

5. **Visualization and Analysis** An interactive user interface is provided to visualize the lineage graph. Users can perform root cause analysis, impact analysis, and data auditing easily using this visualization.

VI. RESULT



- The system was able to keep track of how data moved from one place to another across big data pipelines.
- Without changing any existing pipeline code, metadata was automatically created for every data operation (read, write, transform).
- The lineage graph that was made made it easy to see how datasets, jobs, and users were related.
- Engineers could quickly find the source of data

errors using backward lineage.

- Impact analysis helped find out which reports and jobs were affected before changes were made.
- The time-travel feature let users see how pipeline executions from the past and present in different.
- Visual lineage graphs cut down on debugging time from days to minutes.
- The system made reports that were ready for audits, which helped with compliance and governance.

VII. CONCLUSION

The Data Provenance & Lineage Tracking in Big Data Pipelines project successfully addresses the critical challenge of opacity in modern data engineering. In shifting from manual static document processing to an automated dynamic metadata management system this project bridges the gap between data processing and data governance.

Use of automated collectors together with a graph-based storage engine has been found to be an effective approach for dealing with the non-linear relationships in Big Data processes. This approach puts the power in the hands of the organization in order to gain control of their assets. It ensures the reversal of "Data Swamps" to "Data Lakes" because in the latter it is possible to trace the quality of each piece of data.

The benefits of limited to technicalities of convenience. It forms the cornerstone of regulatory requirements helping organizations to easily implement tough requirements such as GDPR without much human intervention. It helps in improving the efficiency of operations by greatly minimizing the efforts required in debugging as well as analysing its impact. This particular project clearly proves that though Big Data Processing has many complexities yet the management of Big Data Lifecycle can be very transparent visual and intuitive.

REFERENCES

1. Simmhan Y. Plale B. & Gannon D. (2005). "A Survey of Data Provenance in e-Science." *ACM SIGMOD Record* 34(3) 31-36. (This foundational paper establishes the core concepts of provenance tracking).
2. Moreau L. (2010). "The Foundations for Provenance on the Web." *Foundations and Trends in Web Science* 2(2-3) 99-241. (Discusses the Open Provenance Model which influenced the graph design).
3. Zaharia M. et al. (2016). "Apache Spark: A Unified Engine for Big Data Processing." *Communications of the ACM* 59(11) 56-65. (Primary reference for the Spark architecture used in the data collection layer).
4. Ikeda R. & Widom J. (2009). "Panda: A System for Provenance and Data." *Stanford InfoLab Technical Report*. (Provides insights into lineage tracing in warehouse environments).