Dated: 13th Feb 2018

"Data Pipeline Tool Selection Document"

Choices Available:

Pipeline Tools:

1. Apache Kafka
2. Apache NiFi
3. Apache Flume
4. Apache SQOOP
5. Elasticsearch (Logstash & Beats)

Data Stores:

1. HDFS (For un-structured data type)
2. MongoDB / Cassandra / HBase (For semi-structured data)
3. SQL or any other RDBMS software (For structured data)
4. Elasticsearch cluster storage on disk

Event Driven Flow Management Tools:

1. Apache Airflow
2. Apache Oozie

Defining applicability of selected tools in different scenarios:

1. **For '*streaming*' un-structured data type (text file etc.):**
   - **a.** We can use any of **Kafka / NiFi / Flume / Logstash** to ingest data
   - **b.** We will Hadoop Distributed File System **(HDFS)** as Data Store
   - **c.** We can use Airflow / Oozie for event based dataflow management
   - **d.** We can use **Spark / Elasticsearch / Flink / Storm** etc. for real-time processing

2. **For '*batch*' un-structured data type (text file etc.):**
   - **a.** We can use any of **Command Line** / Kafka / **NiFi** / Flume / Logstash
   - **b.** We will Hadoop Distributed File System **(HDFS)** as Data Store
   - **c.** We can use Airflow / Oozie for event based dataflow management
   - **d.** We can use Spark / Elasticsearch / **MapReduce** / Flink etc. for post-processing

3. **For '*streaming*' semi-structured data (CSV, JSON, XML etc.):**
   - **a.** We can use any of Kafka / NiFi / Flume / Logstash
   - **b.** We can use **HDFS / MongoDB / Cassandra / HBase** etc. as Data Store
   - **c.** We can use Airflow / Oozie for event based dataflow management
   - **d.** We can use Spark / **NoSQL Queries / Confluent / Redash** etc. for real-time processing

4. **For '*batch*' semi-structured data (CSV, JSON, XML etc.):**
    a. We can use any of **Command Line** / Kafka / NiFi / Flume / Logstash
    b. We can use **HDFS / MongoDB / Cassandra / HBase** etc. as Data Store
    c. We can use Airflow / Oozie for event based dataflow management
    d. We can use Spark **/ Pig / Hive /** NoSQL Queries / Confluent / Redash etc. for real-time processing

5. **For '*streaming*' structured data:**
    a. We can use Kafka / NiFi / Flume / Logstash etc. to fetch and put data from schema driven relation databases.
    b. We can use **MySQL** or any other RDBMS software as Data Store.
    c. We can use Airflow / Oozie for event based dataflow management
    d. We can use SparkSQL / **SQL Queries** / **Flink** / Confluent / Redash etc. for real-time processing

6. **For '*batch*' structured data:**
    a. We can use Kafka / NiFi / Flume / Logstash / **SQOOP** etc. to fetch and put data from schema driven relation databases.
    b. We can use **MySQL** or any other RDBMS software as Data Store.
    c. We can use Airflow / Oozie for event based dataflow management
    d. We can use SparkSQL / **SQL Queries** / Confluent / Redash etc. for real-time processing

7. **For any other type of data (like directory, S3 etc.):**
    a. We can use **'*hdfs put*'** command from command line, using **direct download**
    b. We can use **Apache NiFi**

8. **Elasticsearch method:**
    a. ELK-stack (Elasticsearch, Logstash, Kibana, Topbeat and X-Pack) can be used in Data Pipeline as a complete package, end to end integration with **Apache NiFi**.
    b. It can also be integrated in Event Flow Management system, if needed, using **Apache Airflow**.

<u>Complete Data Pipeline Implementation Strategy:</u>

1. We will use 3 fronts for data ingestion:
    a. Sqoop (for relational batch data)
    b. Logstash (preferably for text data)
    c. Integrated NiFi, Flume and Kafka Cluster (for any type of data, and will use AVRO for schema validation, if required)

2. We will use 2 tools to manage dataflow operations:
    a. Kakfa (for message passing)
    b. Airflow (for event based action triggering)

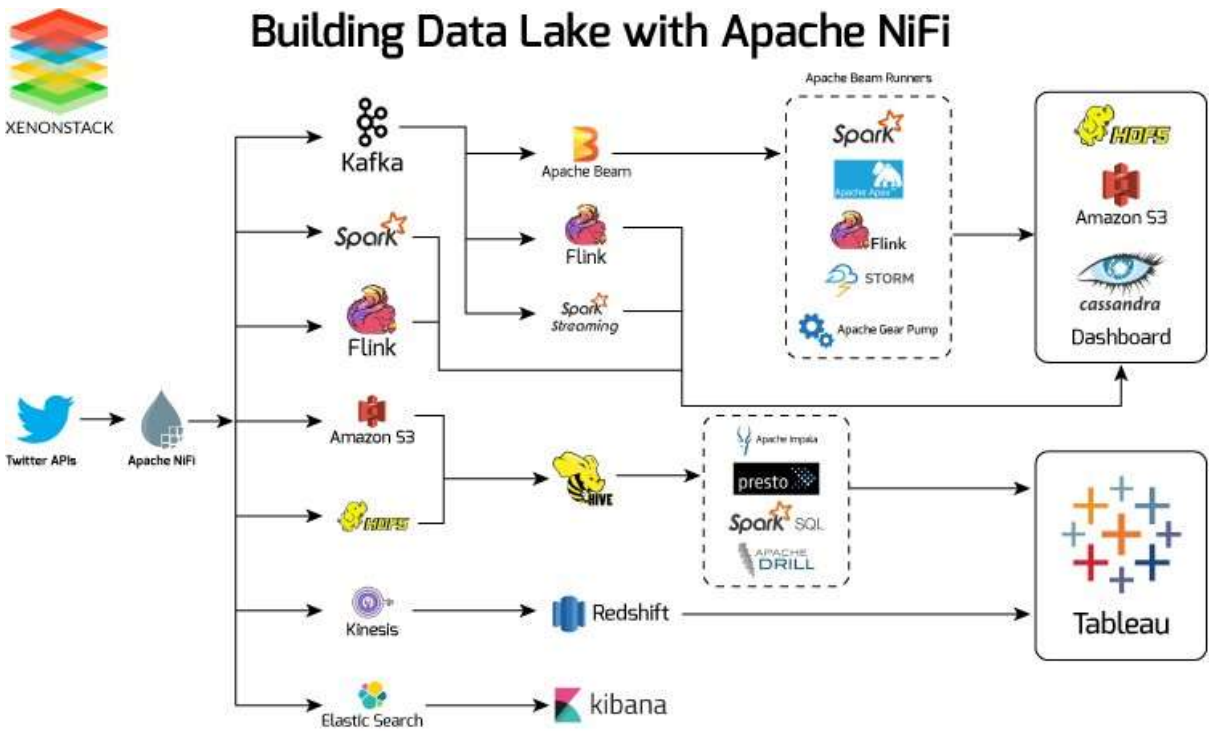
3. We will setup 3 engines for data-processing:
    a. Apache Spark
    b. Elasticsearch
    c. Apache Hive
    d. Apache Pig
    e. Apache Hadoop (MapReduce)
    f. Confluent / Redash or any other (preferred) on the processing engine (Optional) etc.

Comparing similar tools:

NiFi – Data ingestion and Dataflow management

Kafka – Data ingestion and Message passing during dataflow events

Ingestion Tool	Out-of-the-box	Limits	Use Cases
Flume	- Configuration-based - Sources, channels & sinks - Interceptors	- Data loss scenarios when not using Kafka Channel - Data size (KB) - No data replication	- collecting, aggregating, and moving high-volume streaming events into Hadoop.
Kafka	- Back-pressure - Reliable stream data storage - Kafka-Streams - Sources/sinks with Kafka-Connect	- Custom coding often needed - Data size (KB) - Fixed protocol/format/schema	- Streaming data - Messaging - Systems integration - Commit log
NiFi	- Configuration-based UI - Many drag & drop processors - Back-pressure - Prioritized queuing - Data provenance - Flow templates	- Not for CEP or windowed computations - No data replication	- Dataflow management with visual control - Data routing between disparate systems - Arbitrary data size



Cloud vs Local Infrastructure Setup:

1. Local Infrastructure Setup:

1. We need at **least 3 system cluster on-premise** to setup the infrastructure, so that it is configured in **distributed multi-node environment** and hence can be scaled easily in case of more computation needs.
2. Local setup will have a risk of **system failures** and **data loss** as well.

2. Cloud Infrastructure Setup:

1. We do not need to worry about scaling.
2. We need not to configure multi-node system, as AWS single node can be scaled up as per our need.
3. There will be least chance of data-loss or system failure.
4. Data as well as the platform will be available to us anywhere anytime for use.