A Performance Evaluation of Apache Kafka in Support of Big Data Streaming Applications
Paul Le Noac’H, Alexandru Costan, Luc Bougé

To cite this version:

HAL Id: hal-01647229
https://hal.archives-ouvertes.fr/hal-01647229
Submitted on 24 Nov 2017

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
1. Context
- **Stream computing**: a new paradigm enabling real-time Big Data processing through 3 steps
  - Ingestion: Apache Kafka
  - Processing: Apache Spark / Flink
  - Storage: HDFS, Cassandra
- **Ingestion** can be a bottleneck for stream processing

2. Contribution
- Identify the **impact of different parameter** settings on Kafka's overall performance
- **Experiment evaluation** of several configurations and performance metrics of Kafka
- Allow users to **avoid bottlenecks** and achieve good practice for stream processing

3. Kafka Architecture

4. Methodology
- Isolate the performance of each Kafka component
- Separated tests for Producers and Consumers
- Make correlations between configuration parameters, resource usage and performance metrics
- Experiments executed on **Grid5000**
  - Up to 32 nodes (16 cores per node, 28 GB RAM, 10 Gigabit Ethernet)

5. Results
Producer performances when modifying batch size for several number of nodes and a message size of 50B

6. Key metrics

   **Parameters**:
   - Message size
   - Batch size
   - Acquirement strategy
   - Network and disk I/O threads
   - Message replication
   - Hardware

   **Performance Metrics**:
   - Throughput (MB/s, items/s)
   - Latency
   - CPU usage
   - Disk usage
   - Memory usage
   - Network usage

7. Take-aways
- The variation of the batch size shows that there is a range of batches with a better performance.
- When varying the number of nodes in some scenarios: a sudden performance drop (probably due to the internal Kafka synchronizations as well as the underlying network).
- Future work: evaluating reference processing frameworks (Apache Spark and Flink)