Collecting heterogeneous data into a central repository

Daniel Lanza, CERN

Hadoop Users Forum
9th November 2016

Agenda

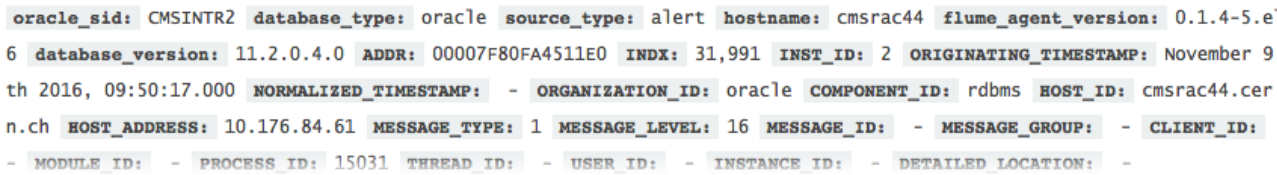
- Project
- What we are collecting
- Custom sources
  - JDBCSource
  - LogFileSource
- Architecture
  - Current to new one
- Tool: Infer Avro schema
- Resources

Project

- Central repository for database audit and logs
- Listener and alert logs to be parsed and stored in the central repository
- Performance metrics for troubleshooting and capacity planning
- Possibility of real-time analytics, offline analytics and visualization
- Reusable open source solution

What we are collecting…

- Tables
  - Audit
  - Metrics



- Log files
  - Listener

```
29-JUL-2016 15:17:34 * (CONNECT_DATA=(SID=DESFOUND)(CID=(PROGRAM=oracle)(HOST=itrac50035.cern.ch)(USER=oracle))) * (ADDRESS=(PROTOCOL=tcp)(HOST=
29-JUL-2016 15:17:38 * service_update * WCERND * 0
Fri Jul 29 15:18:45 2016
29-JUL-2016 15:18:45 * (CONNECT_DATA=(SID=PAYT)(CID=(PROGRAM=RTSDGN@db-50016)(HOST=db-50016)(USER=paytest))) * (ADDRESS=(PROTOCOL=tcp)(HOST=10.1
```

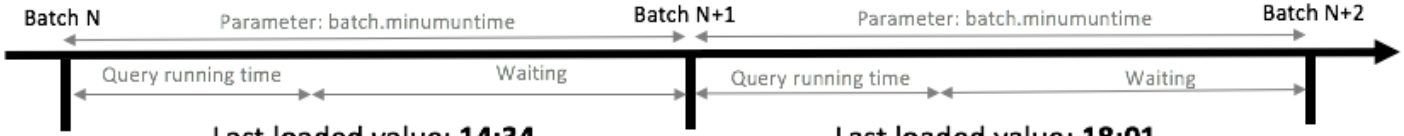
Some numbers

- Coming from 150 servers…
- Per minute
  - 40.000 metrics
  - 5.000 audit events
  - 25.000 listener events

JDBCSource

- A new source which is able to consume from tables
- Any JDBC-compatible database
- Checks periodically for new data
  - A column is used to get only new data: timestamp, ID, sequence number, …
  - If no column is specified, all table is consumed
- Query to be used is generated but can be customized
- A checkpoint file is created
  - Restart or failure will make agent continue from same point
- Duplicated events processor
  - Events with same timestamp? If same event, they are dropped
  - Consuming all table? Only new rows are consumed…

JDBCSource – Why duplicated events processor?



Last loaded value: **14:34**
SELECT * WHERE timestamp > 14:34

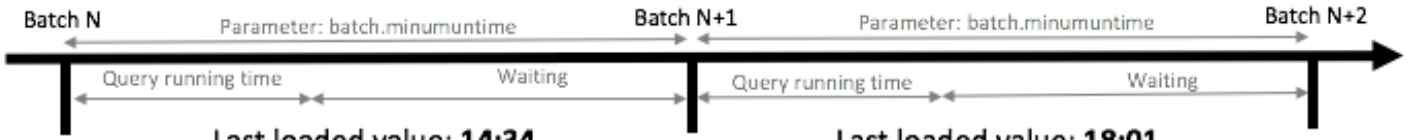
Timestamp	Event type	Event message
13:45	LOGIN	Username: test
14:34	WARN	Disk space limit
18:01	TRACE	Logging out…
18:01	LOGOUT	Username: test

Last loaded value: **18:01**
SELECT * WHERE timestamp > 18:01

Timestamp	Event type	Event message
13:45	LOGIN	Username: test
14:34	WARN	Disk space limit
18:01	TRACE	Logging out…
18:01	LOGOUT	Username: test
18:01	TRACE	User logged out
20:09	WARN	High load

Row in the table before current batch
Missed row
New rows

JDBCSource – Duplicated events processor



Last loaded value: **14:34**
SELECT * WHERE timestamp >= 14:34

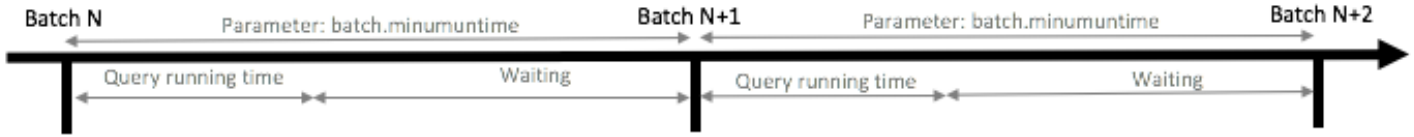
Timestamp	Event type	Event message
13:45	LOGIN	Username: test
14:34	WARN	Disk space limit
18:01	TRACE	Logging out...
18:01	LOGOUT	Username: test

Last loaded value: **18:01**
SELECT * WHERE timestamp >= 18:01

Timestamp	Event type	Event message
13:45	LOGIN	Username: test
14:34	WARN	Disk space limit
18:01	TRACE	Logging out...
18:01	LOGOUT	Username: test
18:01	TRACE	User logged out
20:09	WARN	High load

Row in the table before current batch
Row that collided with previous loaded rows
New rows

JDBCSource – Duplicated events processor



SELECT * FROM table

Event type	Event message
LOGIN	Username: test
WARN	Disk space limit
TRACE	Logging out...
LOGOUT	Username: test

SELECT * FROM table

Event type	Event message
LOGIN	Username: test
WARN	No warning
TRACE	Logging out...
LOGOUT	Username: admin

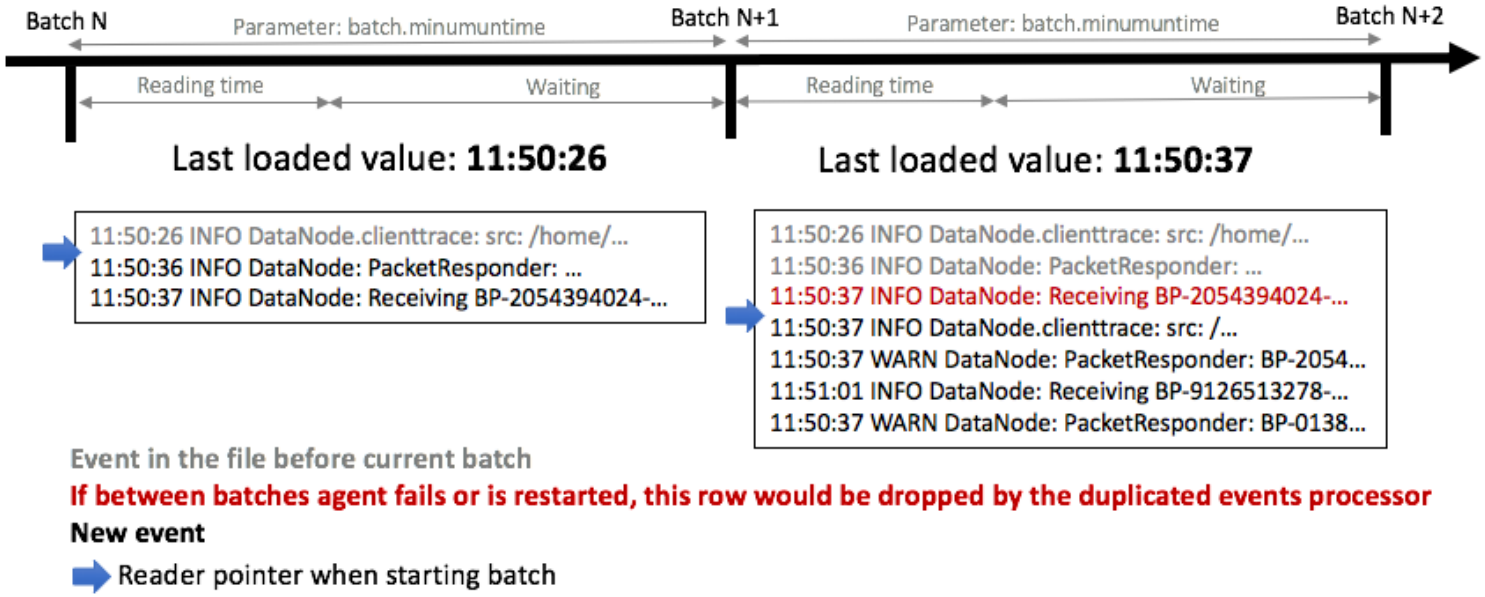
Row that collided with previous loaded rows

New rows

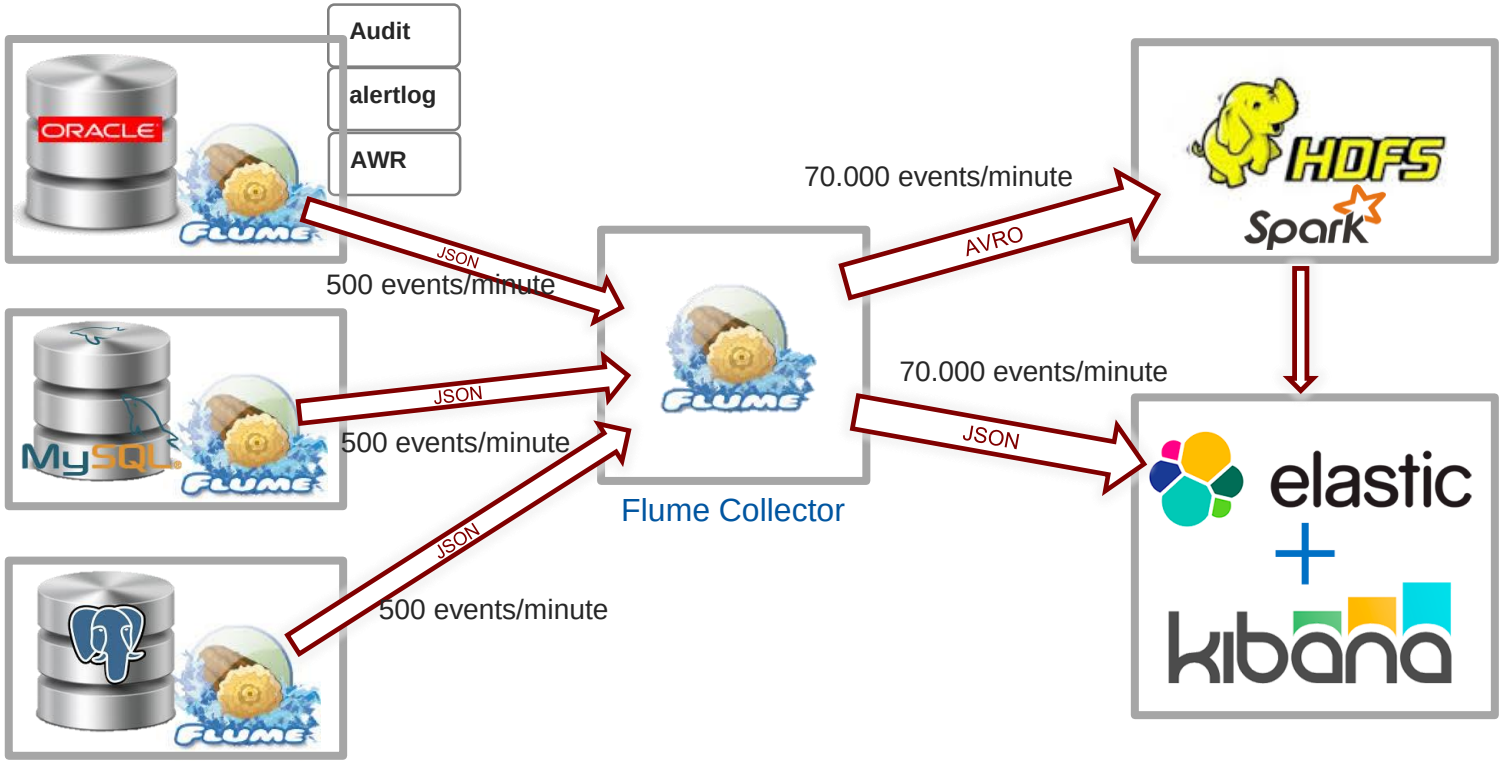
LogFileSource

- A new source which is able to consume log files
- Logic is based on log event timestamp
  - Must contain a timestamp at the beginning of the line (95 % of the cases)
  - Only new events are consumed
- Log file can be rolled out
- A checkpoint file is created with last timestamp
  - Restart or failure of agent will not produce duplicates
- Parse logs before they get into the central repository
- Duplicated events processor
  - Events with same timestamp? If same event, they are dropped
- Note: TailSource coming with Flume 1.7

LogFileSource – Duplicated events processor



Current architecture

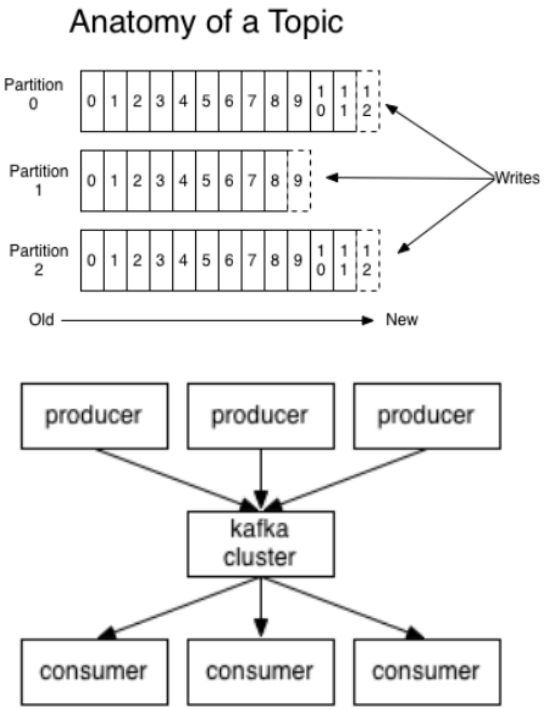


Moving to a new architecture

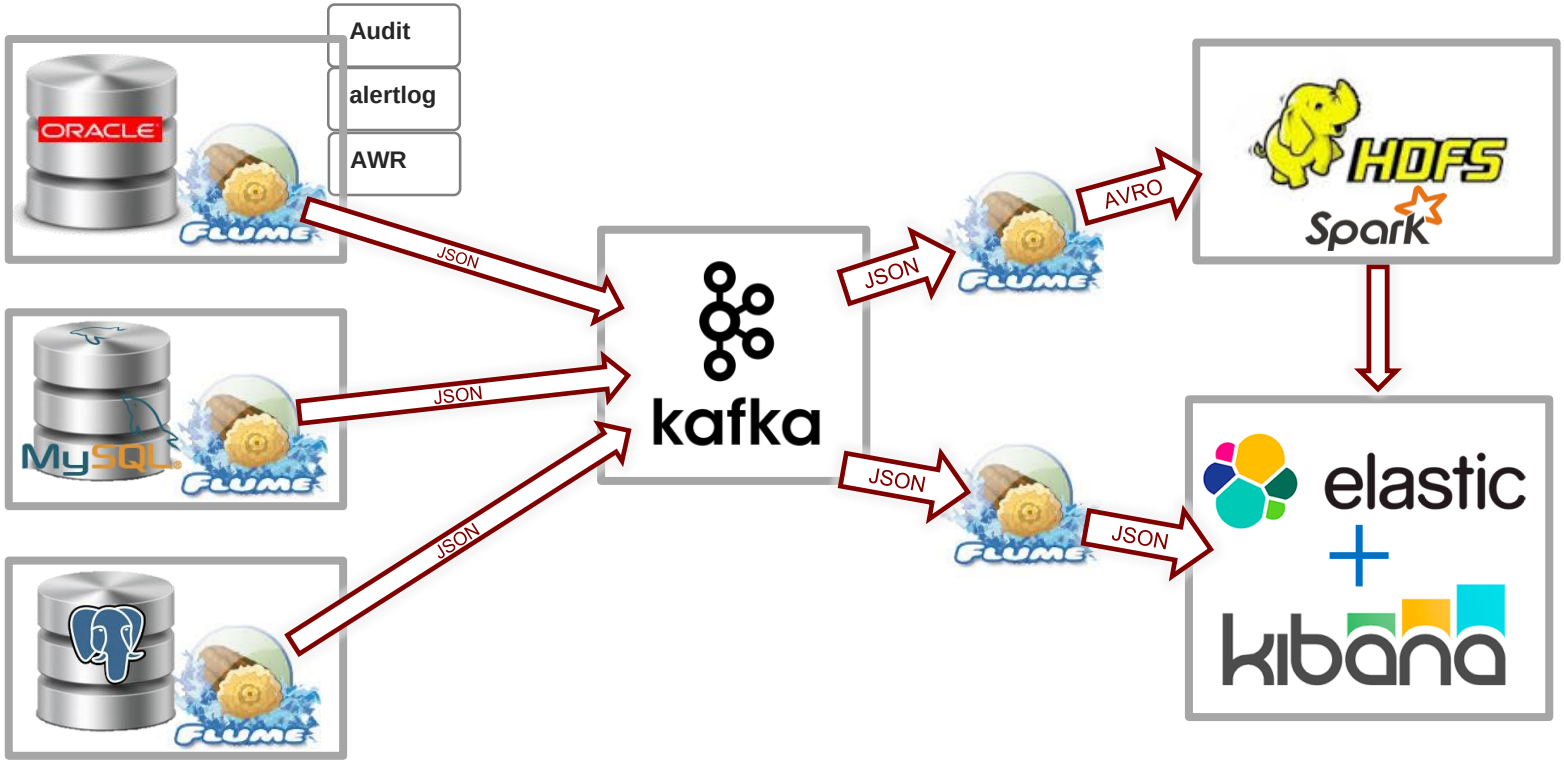
- Problems with current one
  - Failure in any sink will make clients:
    - stop sending data
    - reconnections
    - -> future storm
  - Many agents writing to a single gateway/process
    - Many connections to be kept (memory and resources)
  - Single point of failure
    - Questionable stability -> process killed suddenly because memory consumption
    - We could have replicated agents -> more maintenance and complexity

Moving to a new architecture

- Solution
  - Decouple online and offline layer (Lambda architecture)
  - Apache Kafka
    - Distributed
    - Scalable (partitions)
    - Reliable (replication and acknowledgements)
  - Use IT monitoring infrastructure



New architecture



Tool: infer AVRO schema from table

- Avro schema needs to be created for Avro/Parquet files
- Utility to infer AVRO schema from table metadata

```
bin/infer-avro-schema-from-database -c <Connection URL> -t <TABLE_NAME> -u <USERNSME> -p <PASSWORD> [-dc <DRIVER_FQCN>] [-catalog <CATALOG_NAME>] [-schema <SCHEMA_NAME>]
 -c <CONNECTION_URL>       URL for connecting to database
 -t <TABLE_NAME>           Table from which schema is inferred
 -u <USERNSME>             User to authenticate against database
 -p <PASSWORD>             User's password
 -dc <DRIVER_FQCN>         Fully qualified class name of JDBC driver (default: oracle.jdbc.driver.OracleDriver)
 -catalog <CATALOG_NAME>   Table catalog
 -schema <SCHEMA_NAME>     Table schema
 -help                     Print help
```

Resources

- Documentation: https://database-logging-platform.web.cern.ch/
- Blog entry about developed custom sources: https://db-blog.web.cern.ch/blog/daniel-lanza-garcia/2016-10-custom-flume-sources-ingesting-data-database-tables-and-log-files
- GitLab repository - https://gitlab.cern.ch/db/flume-ng-audit-db

Feedback / Questions