facebook

Operations and Big Data: Hadoop, Hive and Scribe

Zheng Shao 微博：@邵铮9

12/7/2011 Velocity China 2011

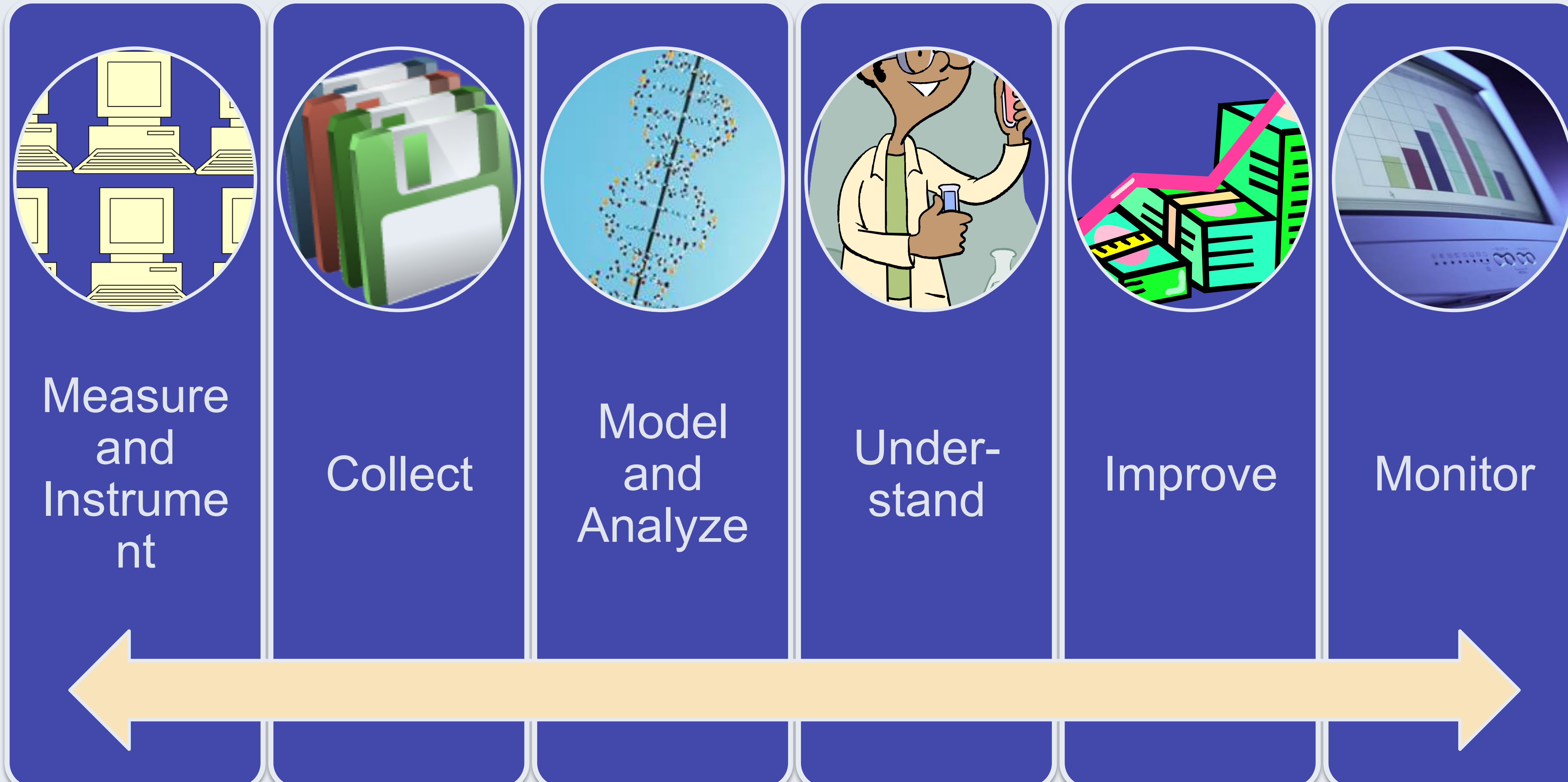
Agenda

1	Operations: Challenges and Opportunities
2	Big Data Overview
3	Operations with Big Data
4	Big Data Details: Hadoop, Hive, Scribe
5	Conclusion

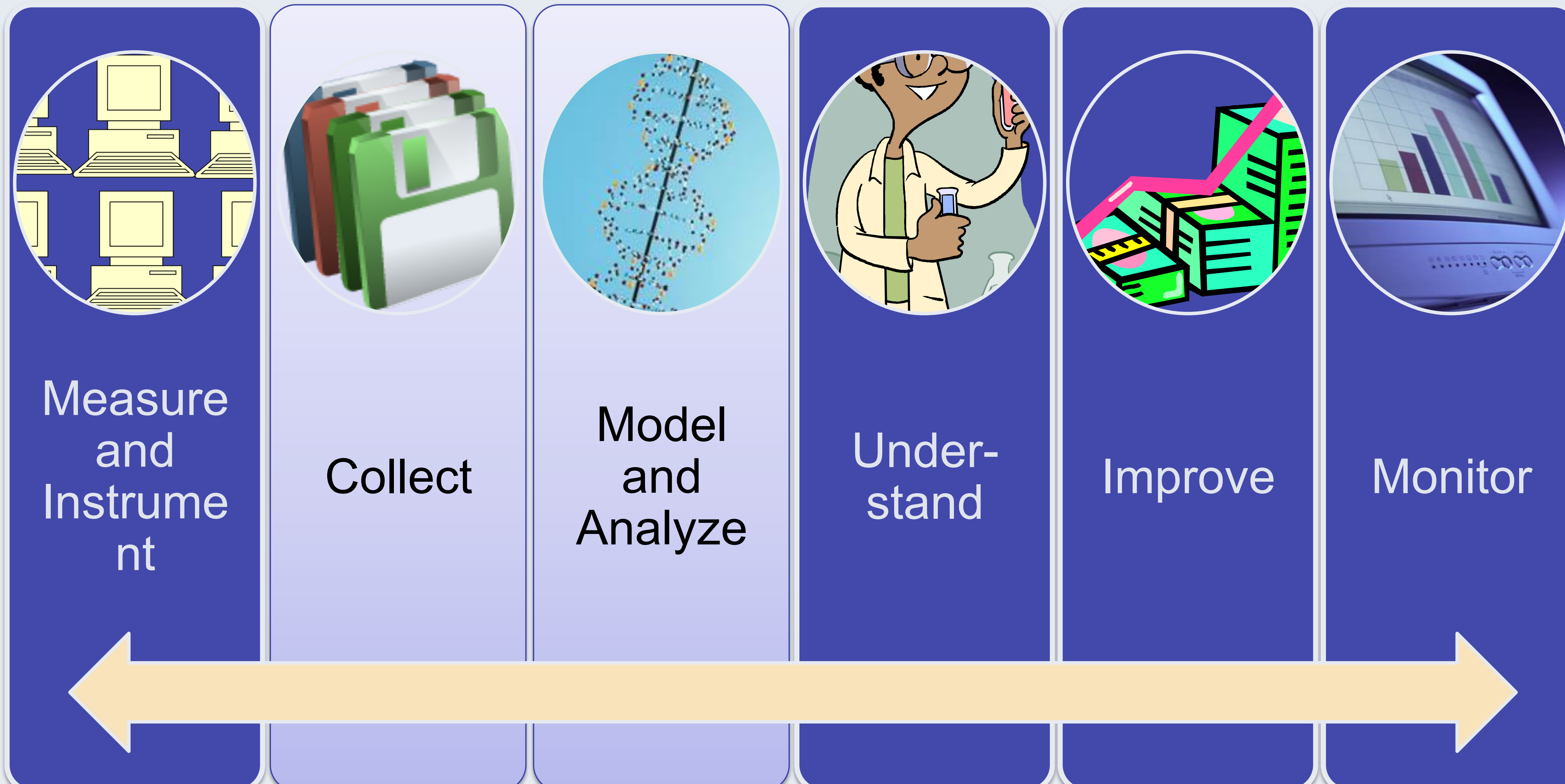
Operations

challenges and opportunities

Operations


Measure and Instrume nt
Collect
Model and Analyze
Under-stand
Improve
Monitor

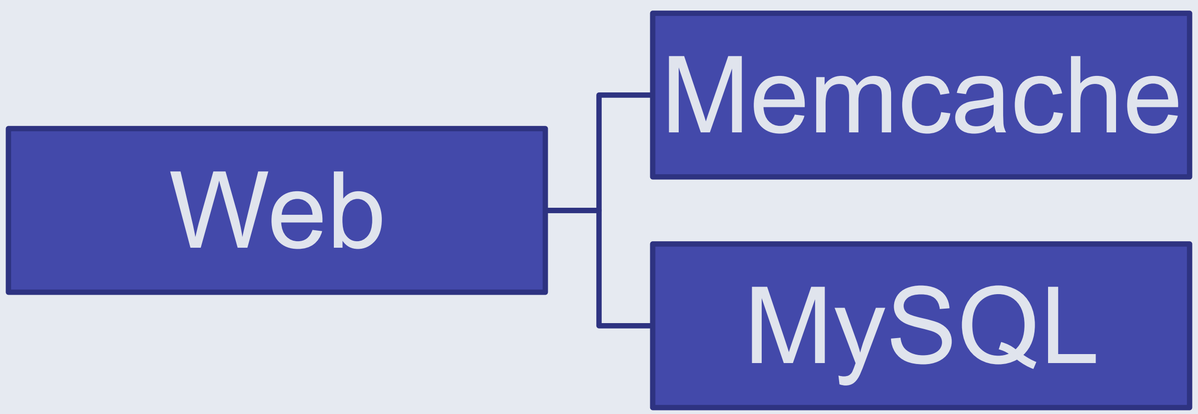
Operations


Measure and Instrume nt
Collect
Model and Analyze
Under-stand
Improve
Monitor

Challenges

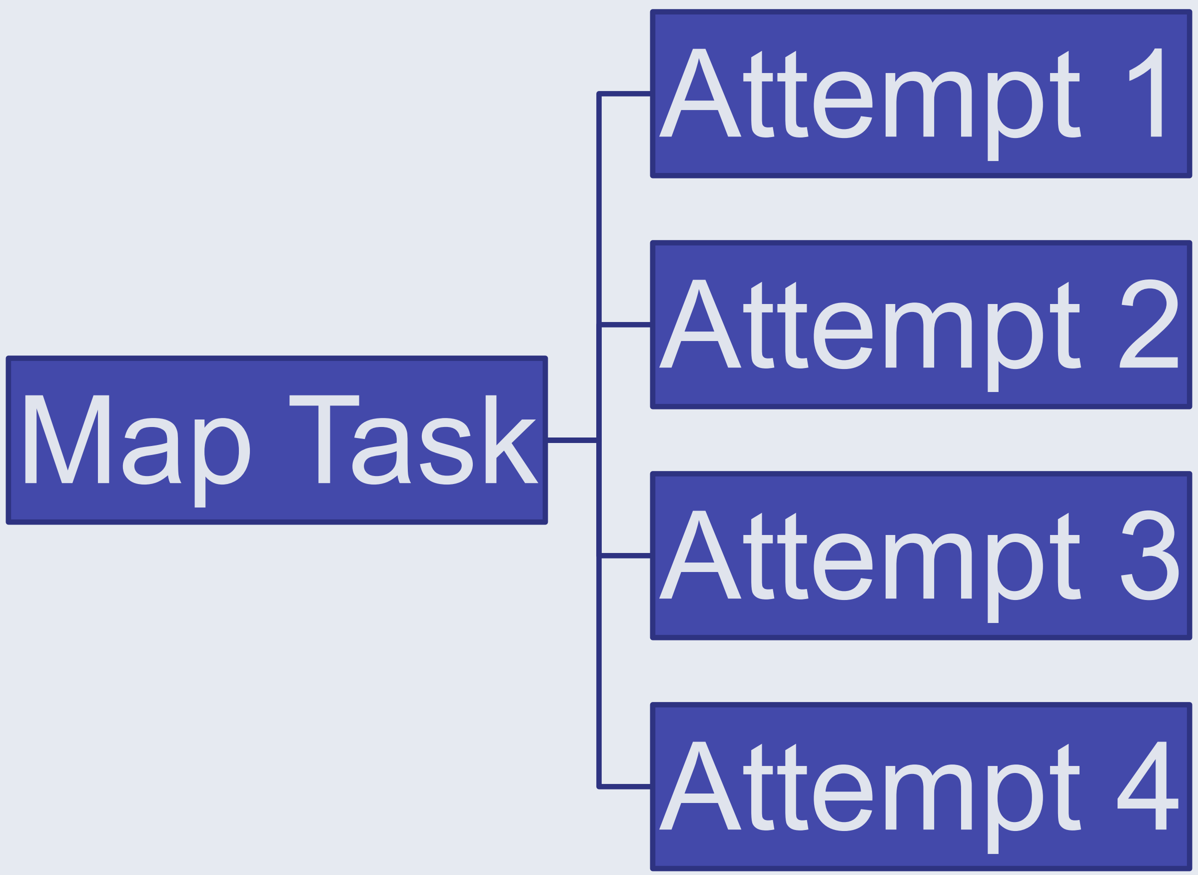
- Huge amount of data
  - Sampling may not be good enough
- Distributed environment
  - Log collection is hard
  - Hardware failures are normal
  - Distributed failures are hard to understand

Example 1: Cache miss and performance



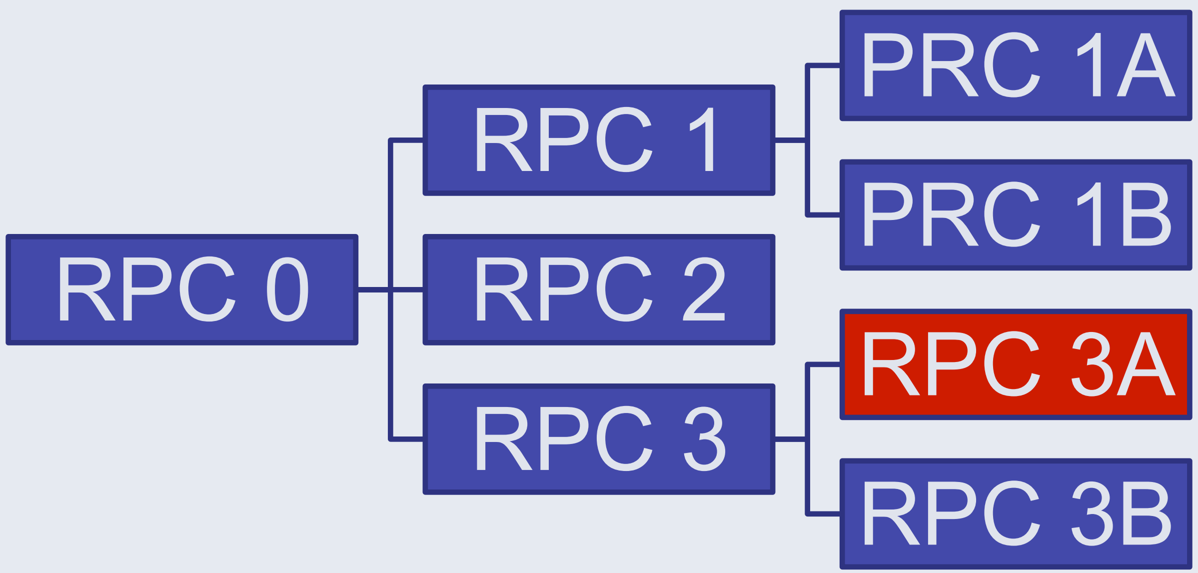
- Memcache layer has a bug that decreased the cache hit rate by half
- MySQL layer got hit hard and performance of MySQL degraded
- Web performance degraded

Example 2: Map-Reduce Retries



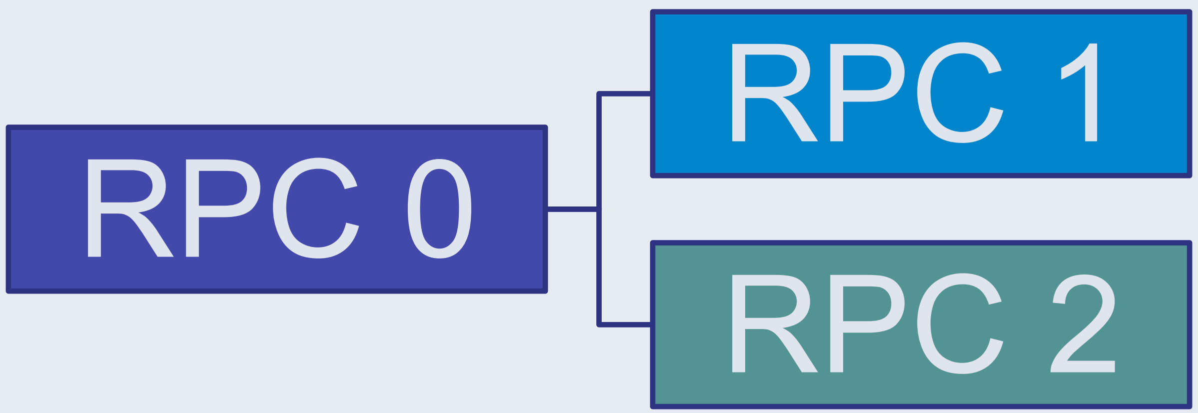
- Attempt 1 hits a transient distributed file system issue and failed
- Attempt 2 hits a real hardware issue and failed
- Attempt 3 hits a transient application logic issue and failed
- Attempt 4, by chance, succeeded
- The whole process slowed down

Example 3: RPC Hierarchy



- RPC 3A failed
- The whole RPC 0 failed because of that
- The blame was on owner of service 3 because the log in service 0 shows that.

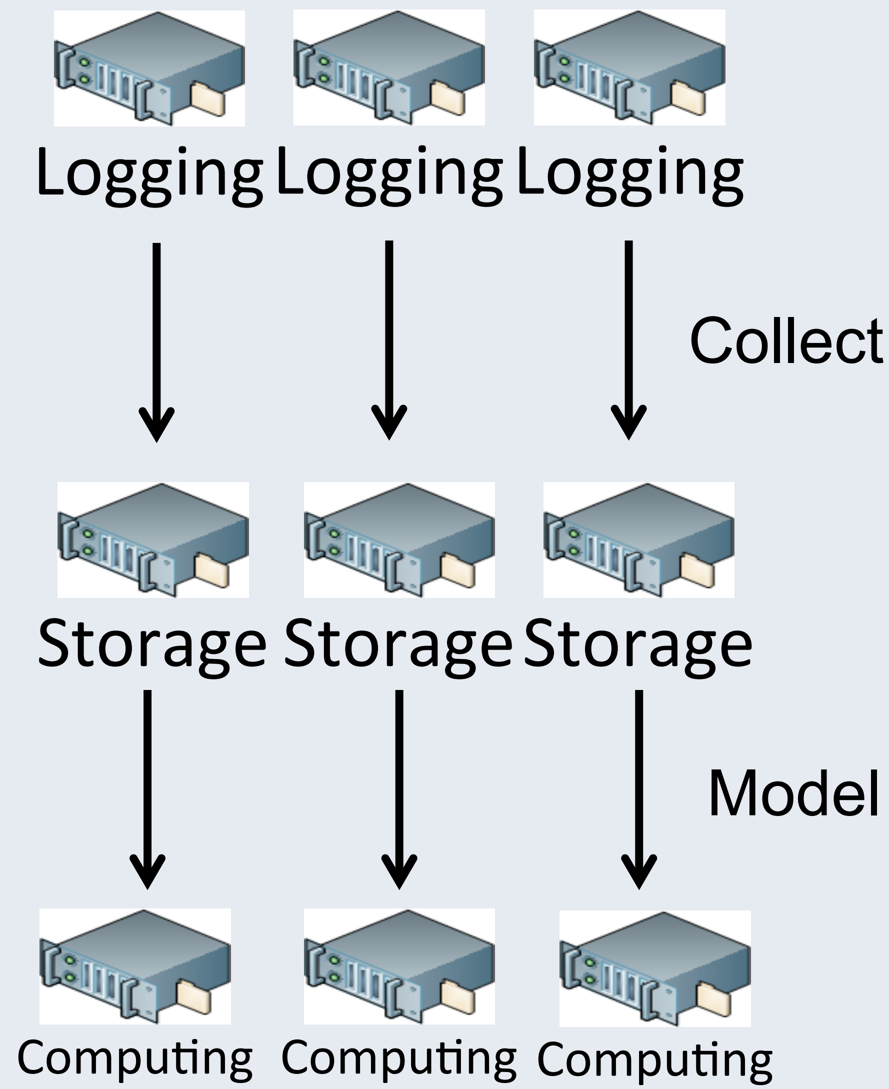
Example 4: Inconsistent results in RPC



- RPC 0 got results from both RPC 1 and RPC 2
- Both RPC 1 and RPC 2 succeeded
- But RPC 0 detects that the results are inconsistent and fails
- We may not have logged any trace information for RPC 1 and RPC 2 to continue debugging.

Opportunities

- Big Data Technologies
  - Distributed **logging** systems
  - Distributed **storage** systems
  - Distributed **computing** systems
- Deeper Analysis
  - Data mining and outlier detection
  - Time-series analysis



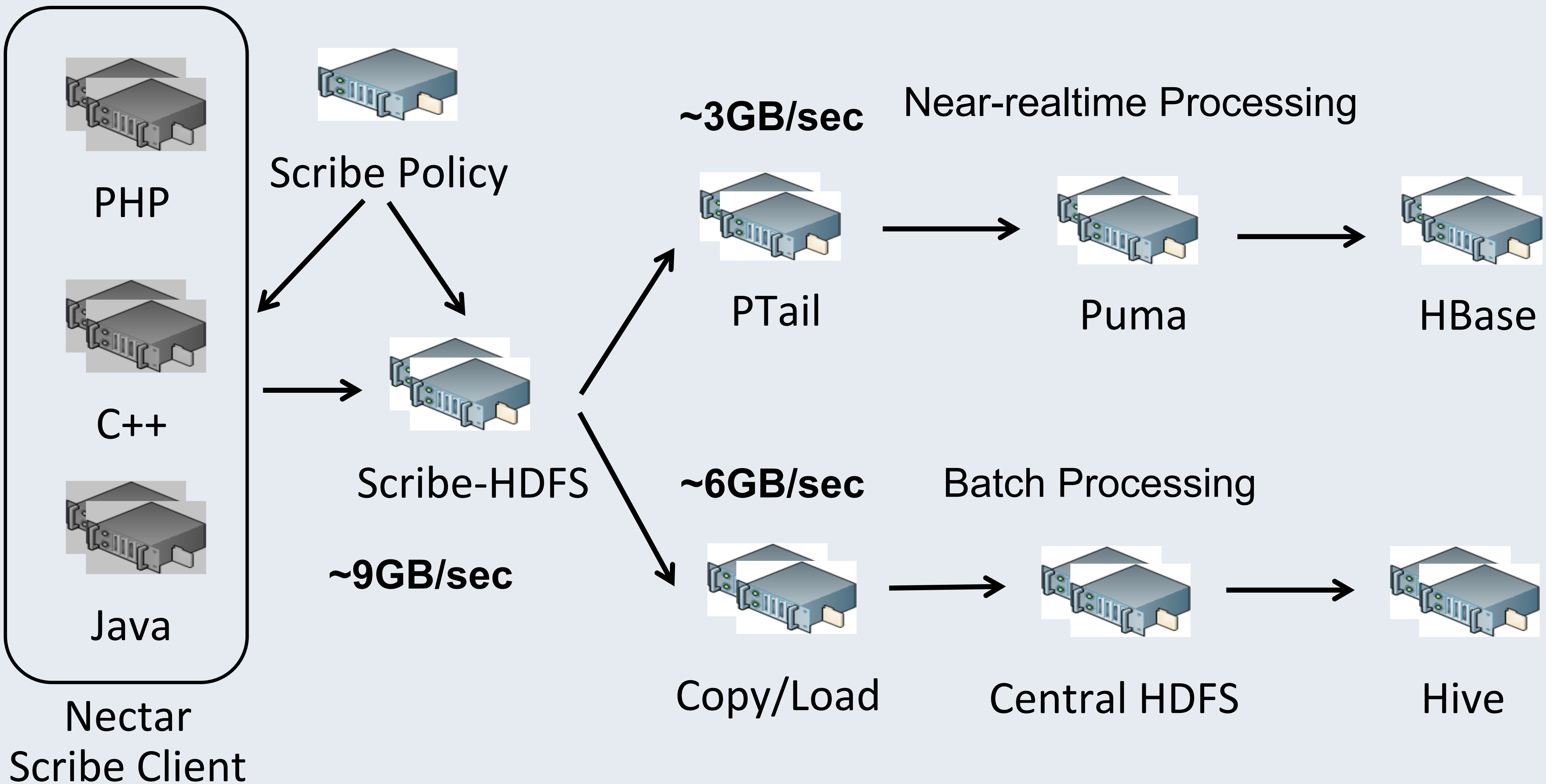
Big Data Overview

An example from Facebook

Big Data

- What is Big Data?
  - Volume is big enough and hard to be managed by traditional technologies
  - Value is big enough not to be sampled/dropped
- Where is Big Data used?
  - Product analysis
  - User behavior analysis
  - Business intelligence
- Why use Big Data for Operations?
  - Reuse existing infrastructure.

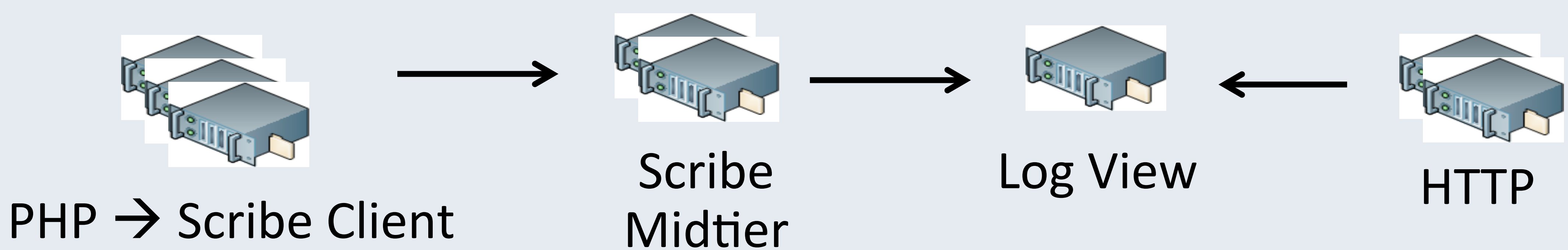
Overall Architecture



Operations
with Big Data

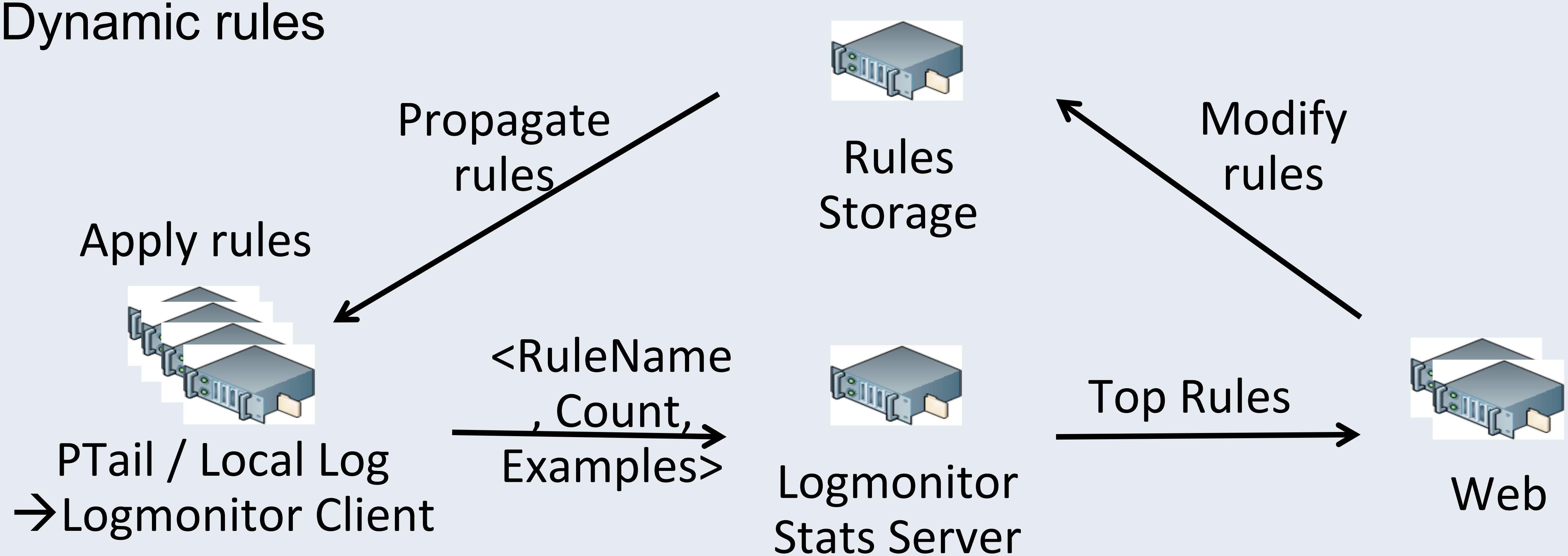
logview

- Features
  - PHP Fatal StackTrace
  - Group StackTrace by similarity, order by counts
  - Integrated with SVN/Task/Oncall tools
  - Low-pri: Scribe can drop logview data



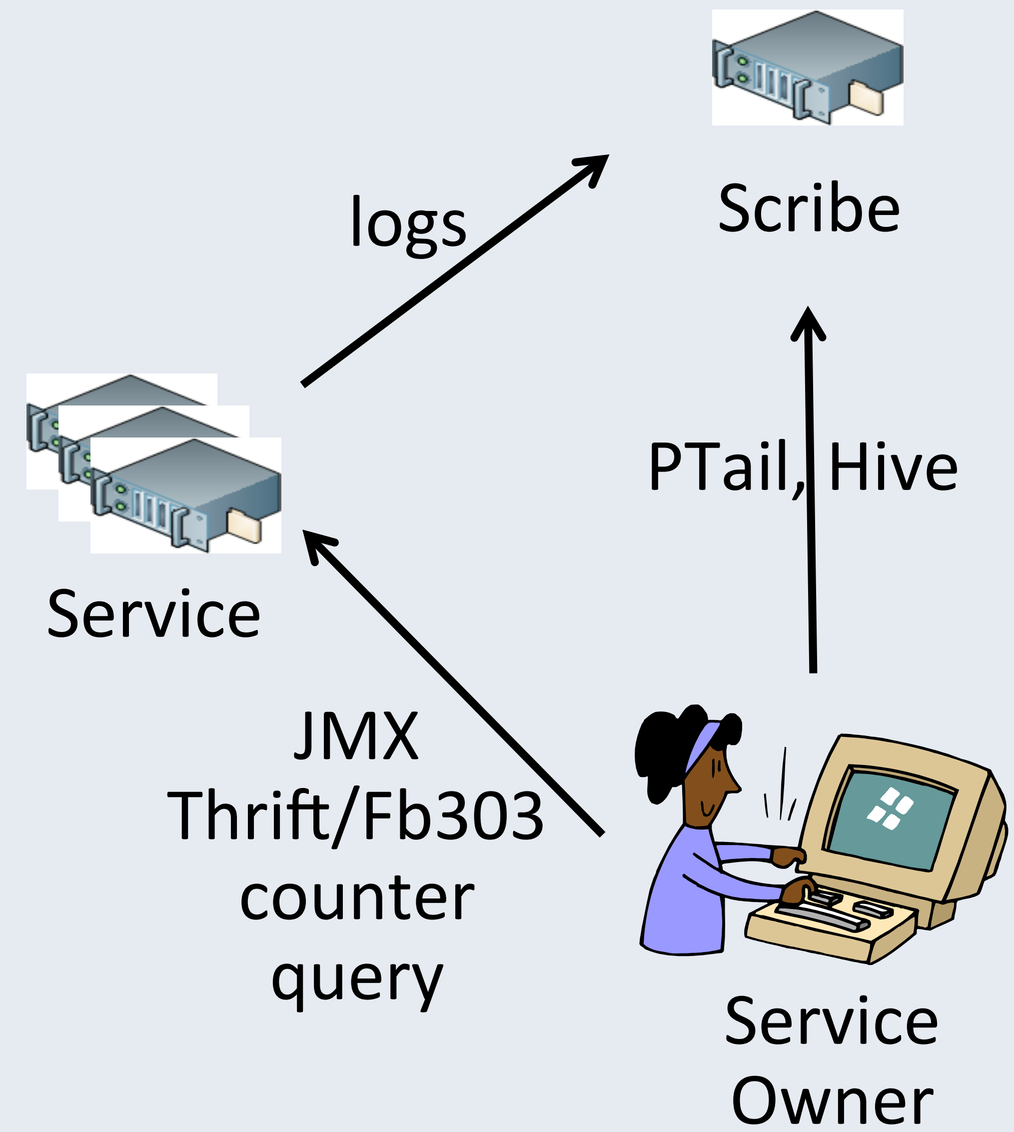
logmonitor

- Rules
  - Regular-expression based: ".*Missing Block.*"
  - Rule has levels: WARN, ERROR, etc
  - Dynamic rules



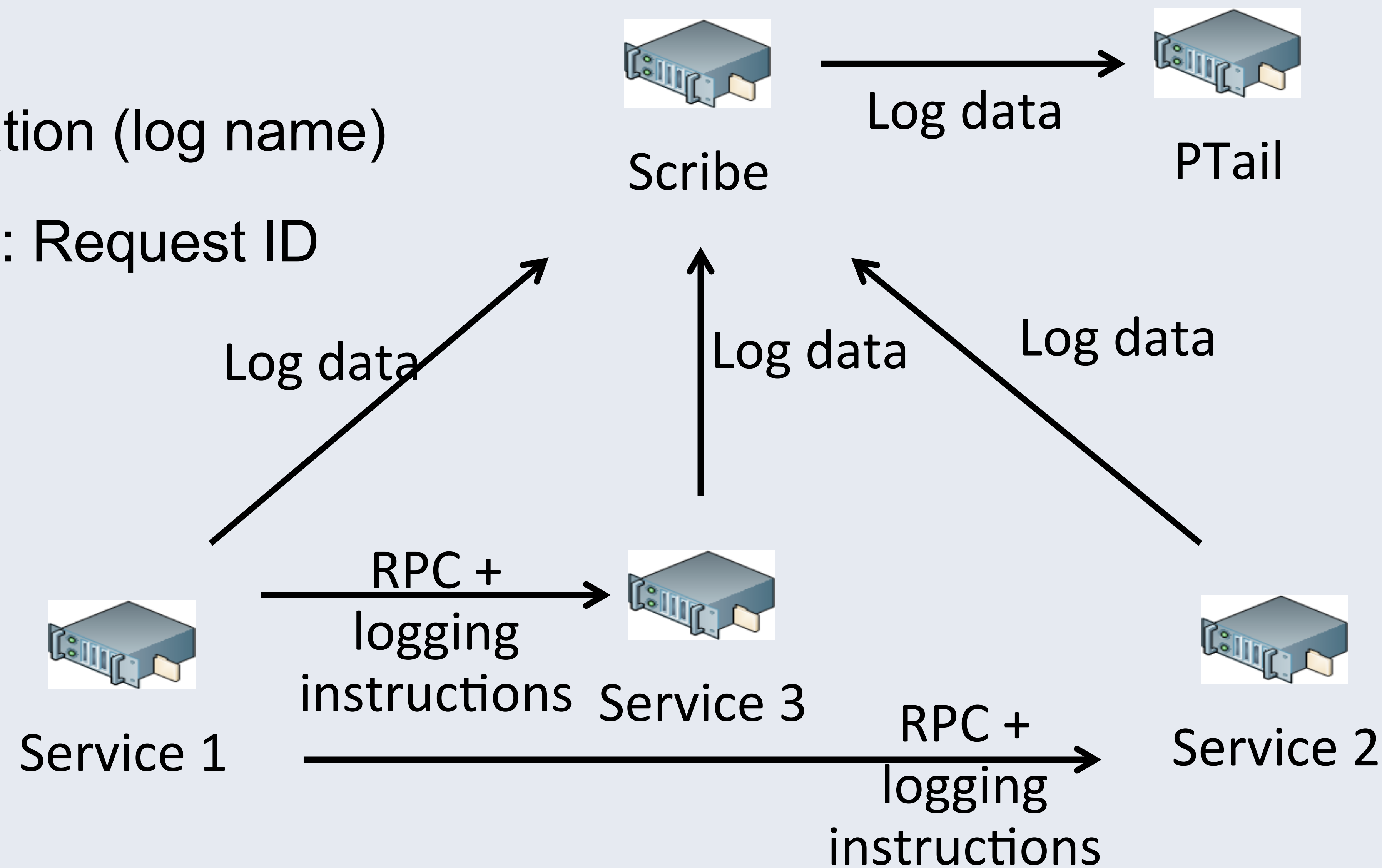
Self Monitoring

- Goal:
  - Set KPIs for SOA
  - Isolate issues in distributed systems
  - Make it easy for service owners to monitor
- Approach
  - Log4J integration with Scribe
  - JMX/Thrift/Fb303 counters
  - Client-side logging + Server-side logging



Global Debugging with PTail

- Logging instruction
  - Logging levels
  - Logging destination (log name)
  - Additional fields: Request ID



Hive Pipelines

- Daily and historical data analysis
  - What is the trend of a metric?
  - When did this bug first happen?
- Examples
  - SELECT percentile(latency, “50,75,90,99”) FROM latency_log;
  - SELECT request_id, GROUP_CONCAT(log_line) as total_log
    FROM trace GROUP BY request_id
    HAVING total_log LIKE "%FATAL%“;

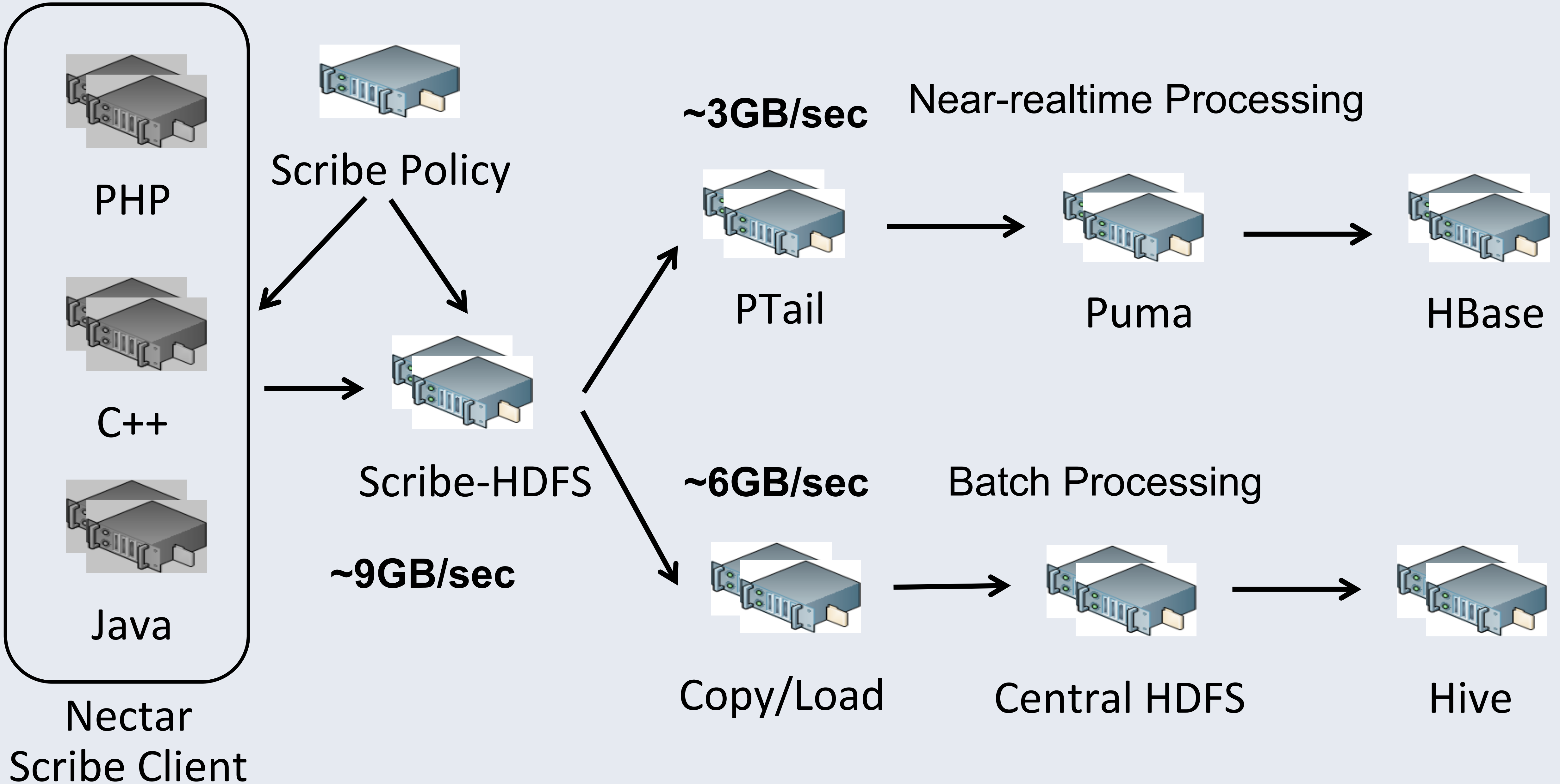
Big Data Details

Hadoop, Hive, Scribe

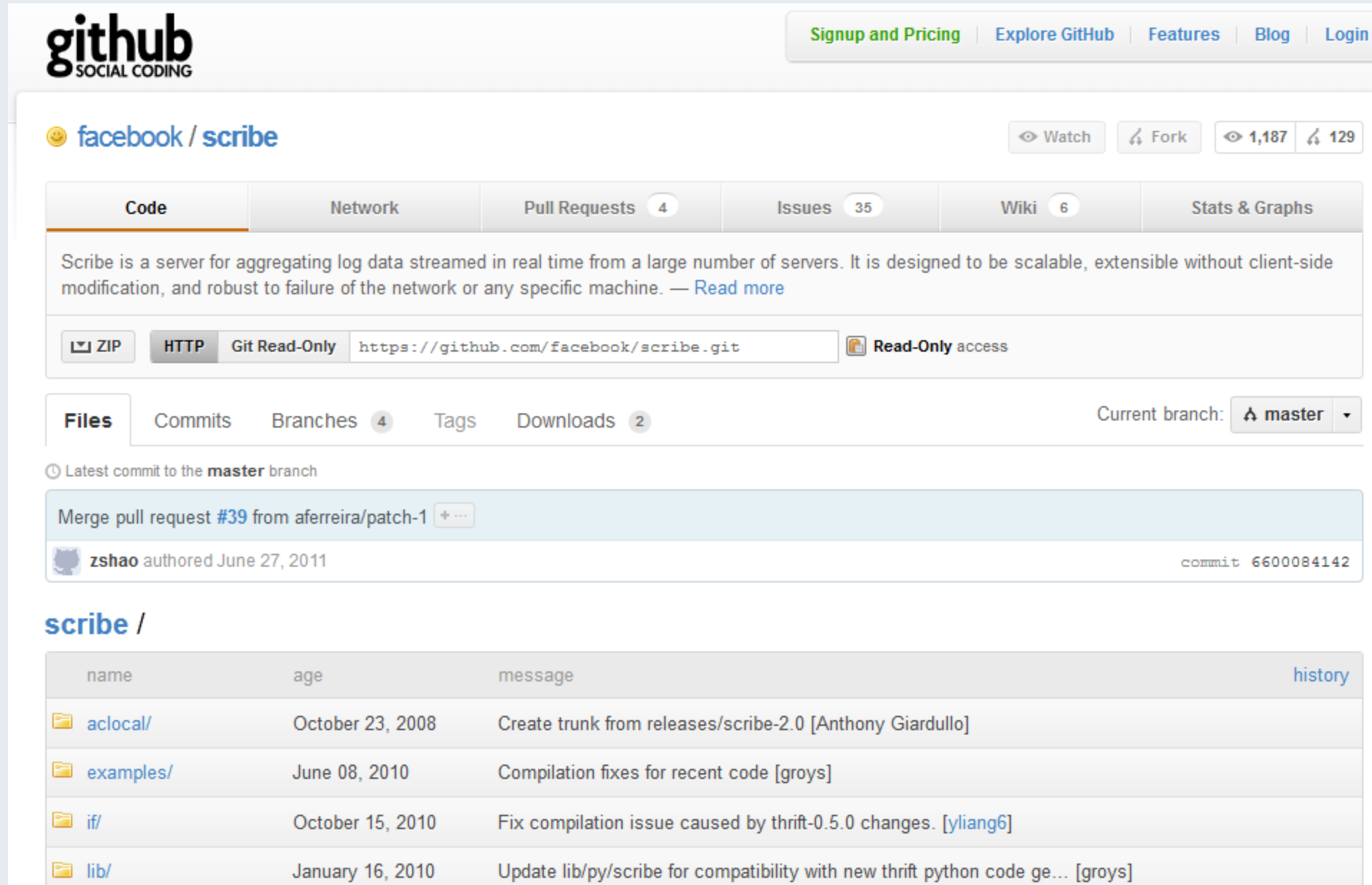
Key Requirements

- Ease of use
  - Smooth learning curve
  - Easy integration
  - Structured/unstructured data
  - Schema evolution
- Scalable
  - Spiky traffic and QoS
  - Raw data / Drill-down support
- Latency
  - Real-time data
  - Historical data
- Reliability
  - Low data loss
  - Consistent computation

Overall Architecture

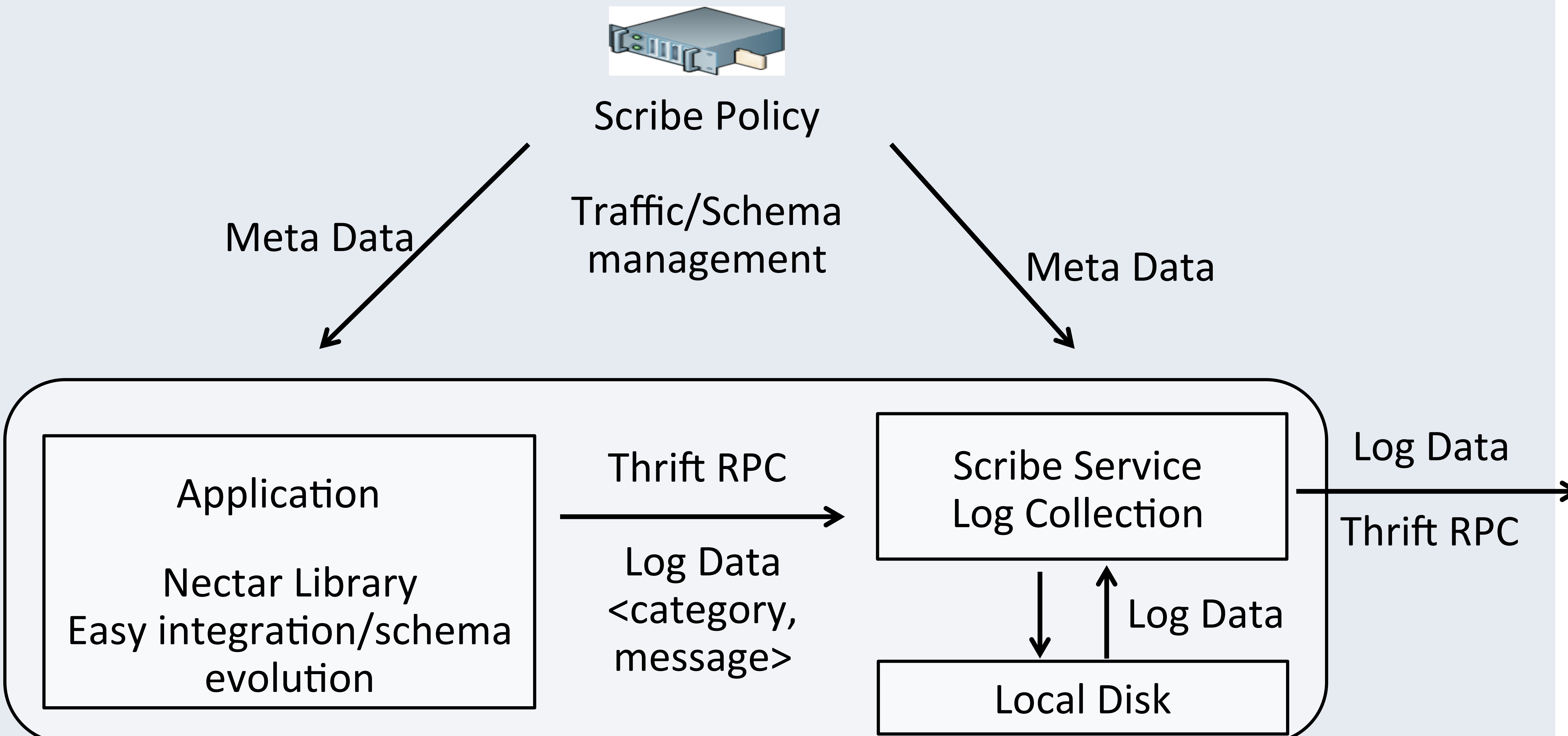


Distributed Logging System - Scribe



- https://github.com/facebook/scribe

Distributed Logging System - Scribe

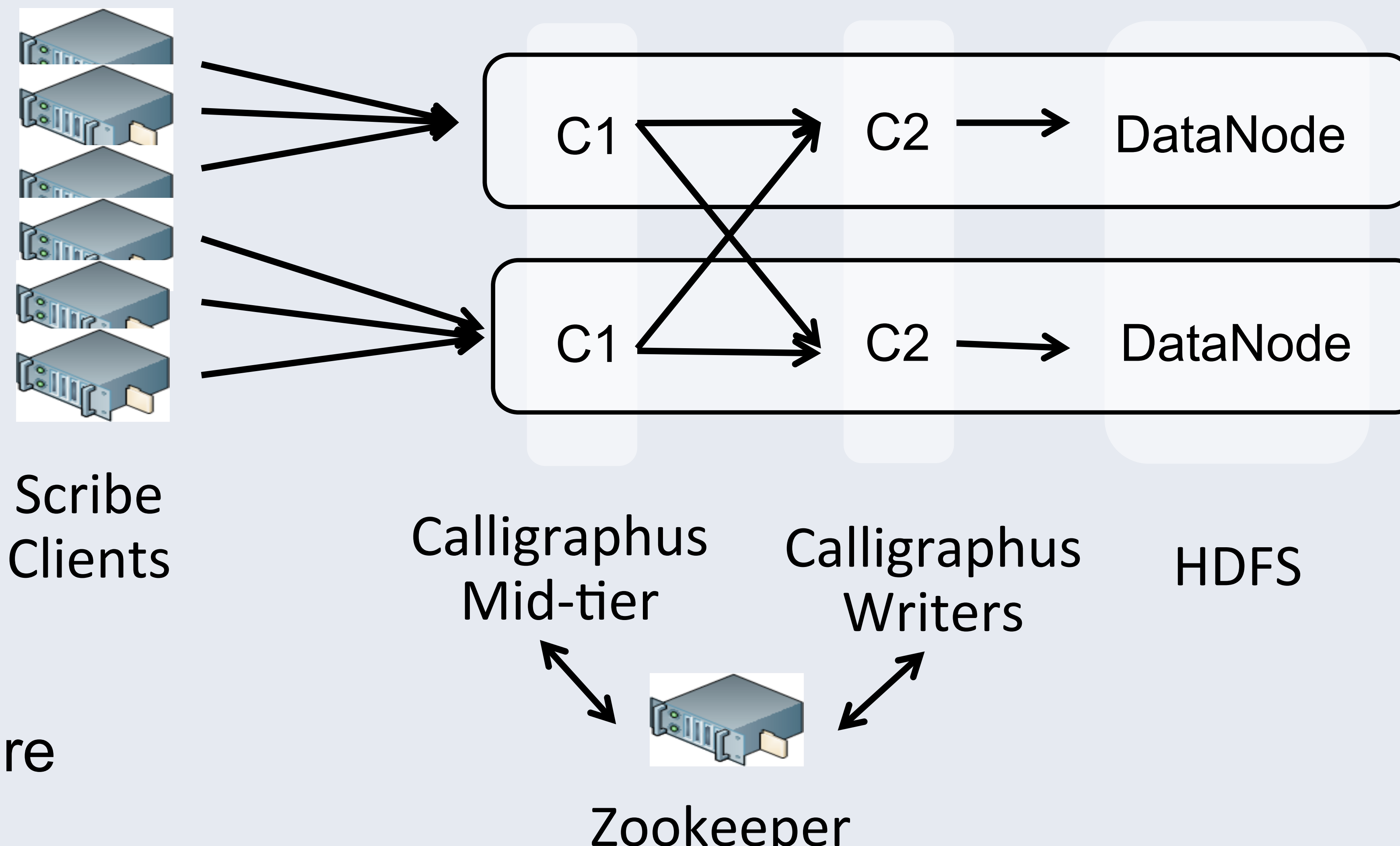


Scribe Improvements

- Network efficiency
  - Per-RPC Compression (use quicklz)
- Operation interface
  - Category-based blacklisting and sampling
- Adaptive logging
  - Use BufferStore and NullStore to drop messages as needed
- QoS
  - Use separate hardware for now

Distributed Storage Systems - Scribe-HDFS

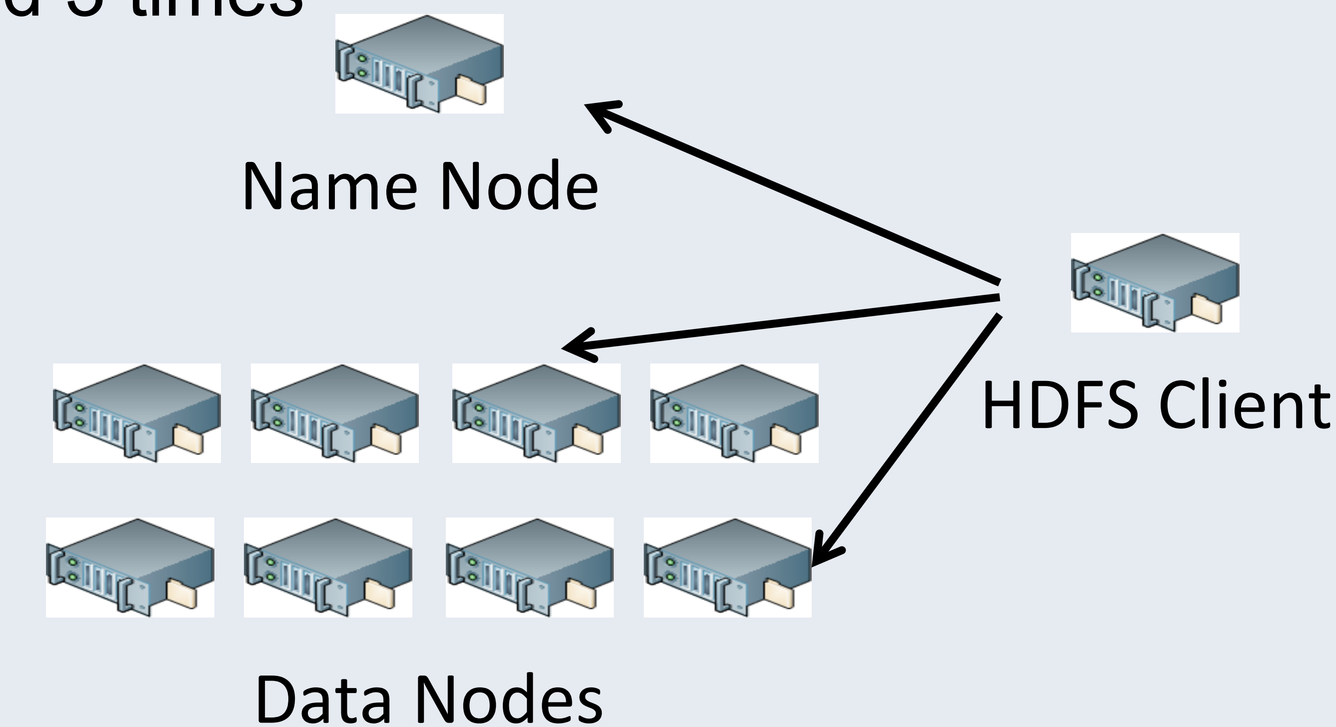
- Architecture
  - Client
  - Mid-tier
  - Writers



- Features
  - Scalability: 9GB/sec
  - No single point of failure (except NameNode)
- Not open-sourced yet

Distributed Storage Systems - HDFS

- Architecture
  - NameNode: namespace, block locations
  - DataNodes: data blocks replicated 3 times
- Features
  - 3000-node, PBs of spaces
  - Highly reliable
  - No random writes
- https://github.com/facebook/hadoop-20

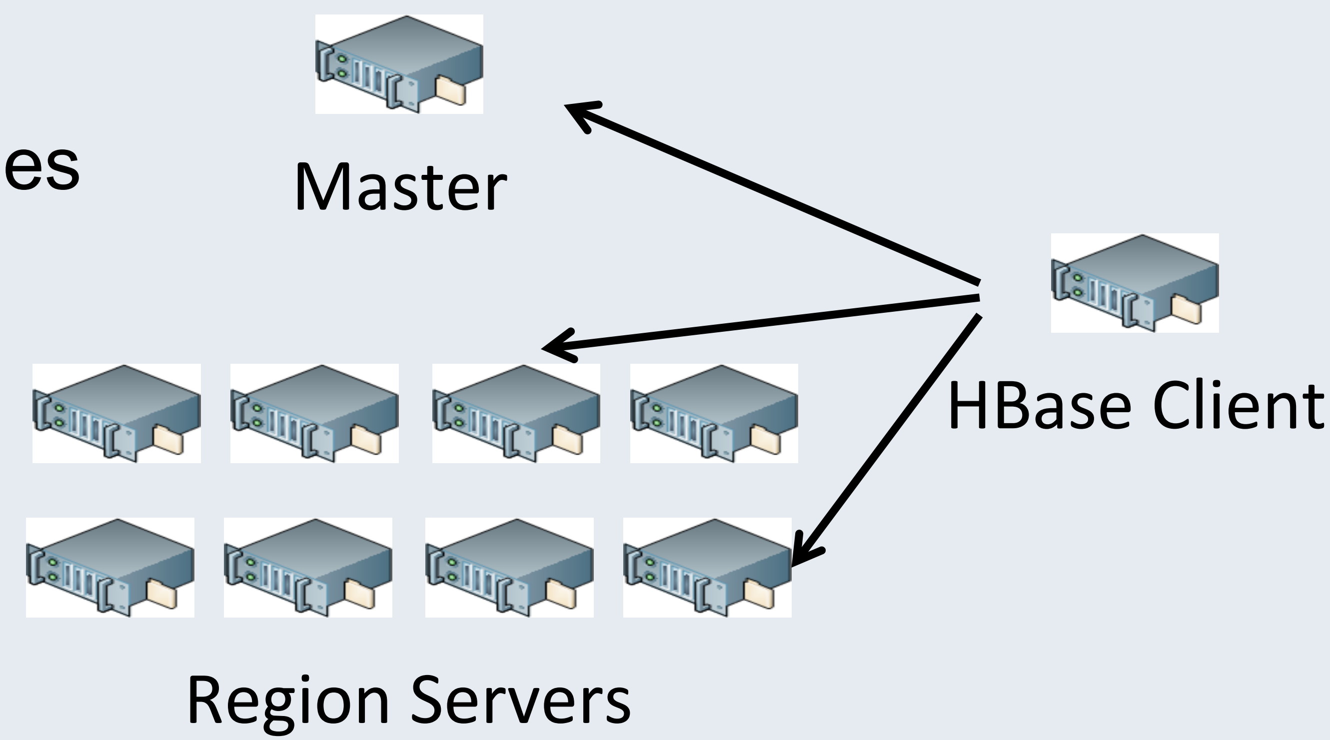


HDFS Improvements

- Efficiency
  - Random read keep-alive: HDFS-941
  - Faster checksum - HDFS-2080
  - Use fadvise - HADOOP-7714
- Credits:
  - http://www.cloudera.com/resource/hadoop-world-2011-presentation-slides-hadoop-and-performance

Distributed Storage Systems - HBase

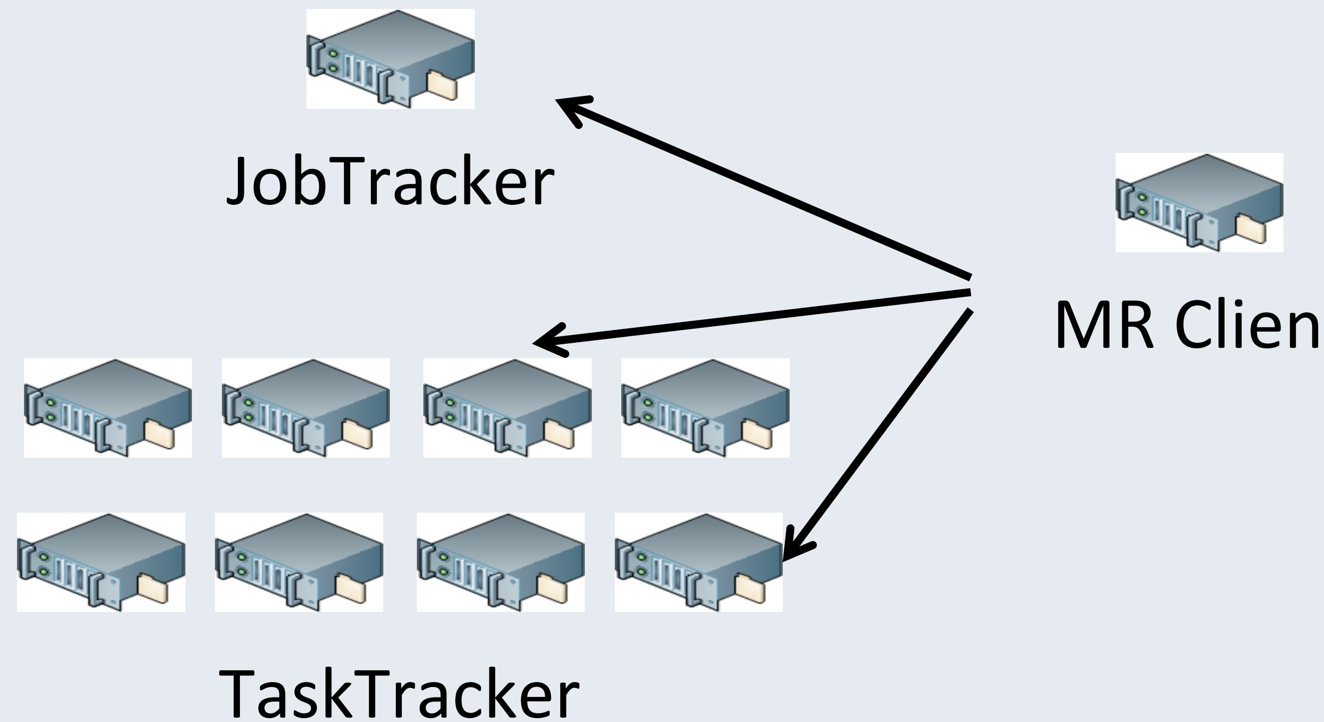
- Architecture
  - <row, col-family, col, value>
  - Write-Ahead Log
  - Records are sorted in memory/files



- Features
  - 100-node.
  - Random read/write.
  - Great write performance.
  - http://svn.apache.org/viewvc/hbase/branches/0.89-fb/

Distributed Computing Systems – MR

- Architecture
  - JobTracker
  - TaskTracker
  - MR Client



- Features
  - Push computation to data
  - Reliable - Automatic retry
  - Not easy to use

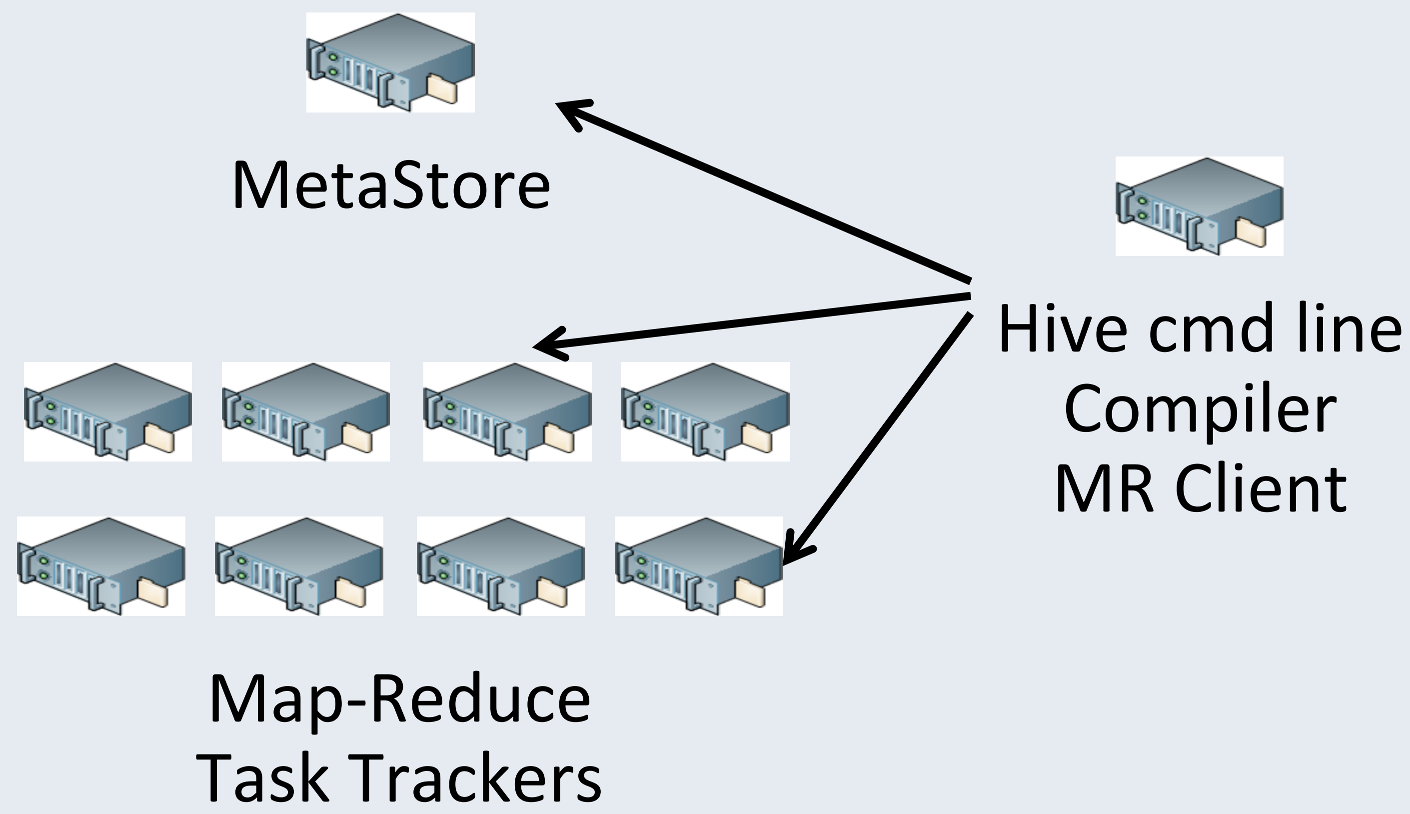
MR Improvements

- Efficiency
  - Faster compareBytes: HADOOP-7761
  - MR sort cache locality: MAPREDUCE-3235
  - Shuffle: MAPREDUCE-64, MAPREDUCE-318

- Credits:
  - http://www.cloudera.com/resource/hadoop-world-2011-presentation-slides-hadoop-and-performance

Distributed Computing Systems – Hive

- Architecture
  - MetaStore
  - Compiler
  - Execution
- Features
  - SQL → Map-Reduce
  - Select, Group By, Join
  - UDF, UDAF, UDTF, Script

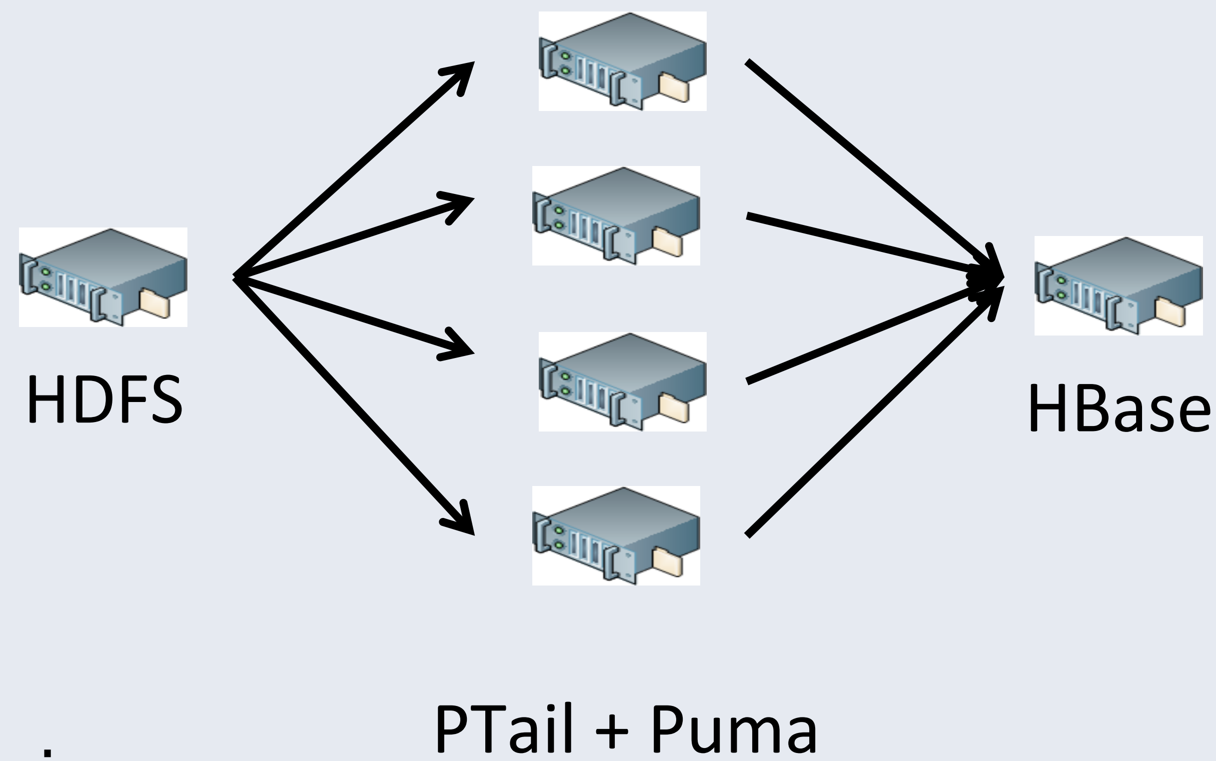


Useful Features in Hive

- Complex column types
  - Array, Struct, Map, Union
  - CREATE TABLE (a struct<c1:map<string,string>,c2:array<string>>);
- UDFs
  - UDF, UDAF, UDTF
- Efficient Joins
  - Bucketed Map Join: HIVE-917

Distributed Computing Systems – Puma

- Architecture
  - HDFS
  - PTail
  - Puma
  - HBase



- Features
  - StreamSQL: Select, Group By, Join
  - UDF, UDAF
  - Reliable – No data loss/duplicate

Conclusion

Big Data can help operations

Big Data can help Operations

- 5 Steps to make it effective:
  - Make Big Data easy to use
  - Log more data and keep more sample whenever needed
  - Build debugging infrastructure on top of Big Data
  - Both real-time and historical analysis
  - Continue to improve Big Data

facebook