Big Data Analysis of Cloud Storage Logs using Spark

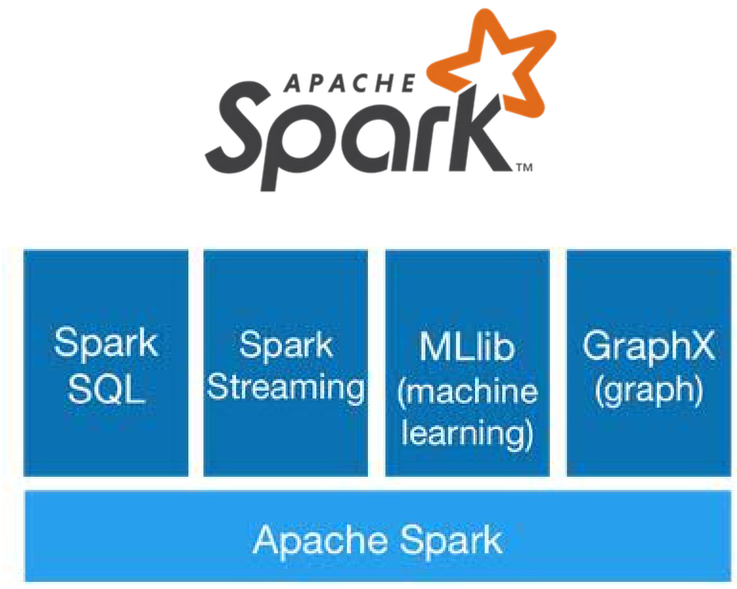
Shelly Garion, Hillel Kolodner, Allon Adir, Ehud Aharoni, Lev Greenberg

IBM Research - Haifa, Israel
{shelly, kolodner, adir, aehud, levg}@il.ibm.com

Systor2017 May 22 – 24 Haifa, Israel
10 The 10th ACM International Systems and Storage Conference

Motivation

- Investigate the logs of an operational cloud object store service to understand how it is used
- Requires going over very large amounts of historical data (e.g., PBs of records) collected over long periods
- Existing tools, such as Elasticsearch, Logstash, or Kibana are good for presenting short-term metrics, but cannot perform advanced analytics
- Apache Spark is good for log analysis and advanced analytics, but we still need to use it smartly
- Our techniques include sampling, smart grouping and aggregation, and the use of machine learning methods targeted at log data



Use-case 1: Latency analysis

- Problem: Identify time frames in which the performance decreased
- Challenge: Impractical to collect all the latencies, sort them, and calculate the exact percentiles
- Methods:
  - Focus on HEAD operations
  - Divide latencies into a histogram using the "Map/Reduce" method

```
// Main:
val logFiles =
sc.textFile ("hdfs:///logdata/logdatafile*.gz")
val LatencyHEADobject = logFiles.map(_split(" ")).
map(ProcessLogLine).filter(line => line._1 ==
"HEAD object")
val LatencyHistogram =
LatencyHEADobject.map(LatencytoBuckets).
reduceByKey((a,b)=>a+b)

// Functions:
def ProcessLogLine(line: Array[String]) = {
val operation = .. // string
// contains the fields indicating the operation type
val time = .. // string
// contains the fields indicating the request time
// (either week, day, hour, 10 minutes, minute)
val latency = .. // double
//the field indicating the latency of the request
(operation, time, latency)
}

def LatencytoBuckets(line: (String,String,Double))= {
val time = line._2
val latency = line._3
val loglatency = math.log(latency*1000)
val bucket = (if (loglatency > 0) loglatency.toInt
else 0)
((time, bucket),1L)
} //the graph in shows buckets 1 to 6.
```

Algorithm for latency analysis

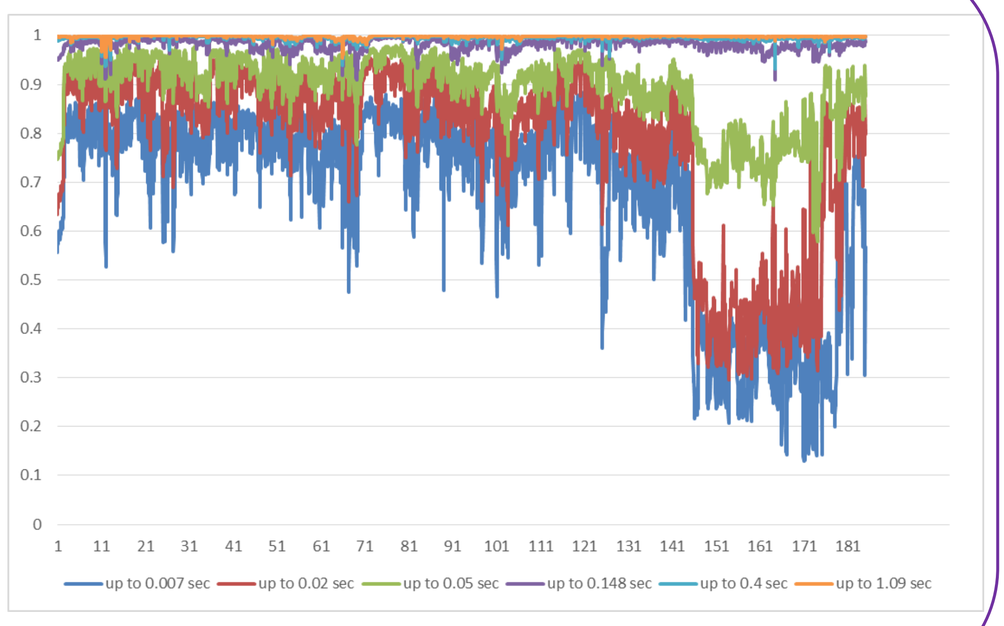


Figure 1: Distribution of latency of HEAD object

Use-case 2: Archiving potential

- Problem: Estimate the potential for archiving, e.g., estimate the number of candidate objects and the expected archive size
- Challenge: Impractical to compile information for all objects that have ever been created, used, rewritten, or erased
- Methods:
  - Take a random sample of the objects
  - Two passes over the data – daily reduction and a final analysis on the daily summaries

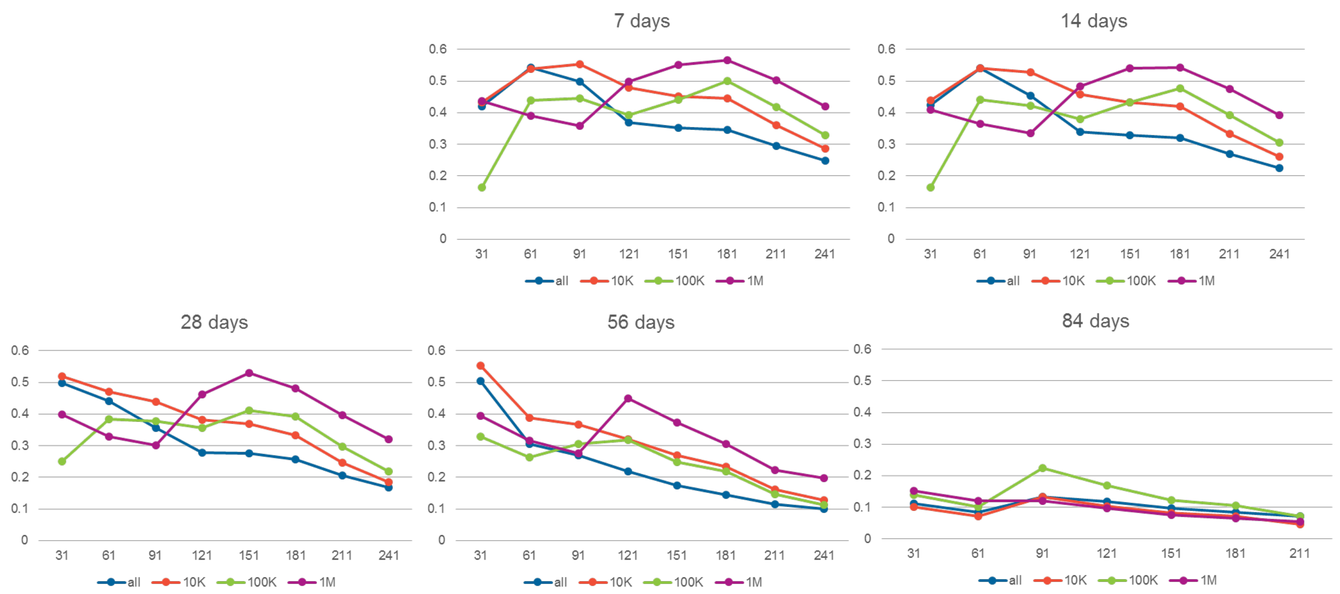


Figure 2: The probability that an object will be touched again if it has not been touched for T days (for T=7, 14, 28, 56, 84) as a function of the day number

Use-case 3: Anomaly detection

- Problem: Detect security threats and anomalies in object accesses
- Challenge: Large volume of operations on an object store and very large number of objects
- Methods:
  - Train a model of "normal" customer behavior over long time spans
  - Detect activities with significant deviations from the trained models and report alerts

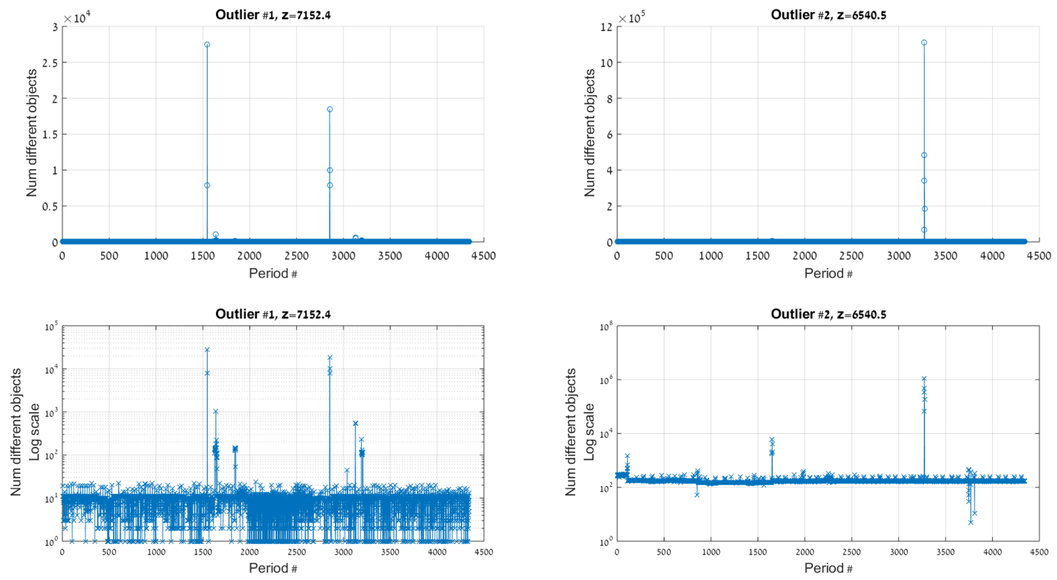


Figure 3: Abnormalities in access to objects for two accounts with high Z-scores