A Big Data Framework for Cloud Monitoring

Saeed Zareian, Marios Fokaefs, Hamzeh Khazaei, Marin Litoiu, Xi Zhang
Department of Electrical Engineering and Computer Science
York University
Toronto, ON, Canada
fokaefs@yorku.ca
Agenda

• Motivation
• Problem
• K-Feed Monitoring Framework
• Experiments
• Discussion
Self-Adaptive Software Systems on the Cloud

MAPE-K Loop

Monitored data need to be:
- Accurate
- On time
- Gathered fast in short intervals (e.g. <1 minute)
- Meaningful for the Managed Resource
- Meaningful for the adaptation algorithms.
Monitored data is Big Data!
Monitored data: Characteristics and Challenges

• 5 monitors X 10 metrics X 120 apps X 6 VMs X 120 mins = 4.32 M measurements in two hours. (Volume)

• The volatility of cloud software dictates adaptive actions as frequent as every minute. (Velocity)

• Data can come from a number of sources, concerning a number of resources and metrics, in different frequencies and intervals, causing some times synchronization problems. (Variety/Veracity)

• We need to employ a NoSQL, high-throughput to support monitoring-as-a-service on the cloud.
K-Feed Monitoring Framework

- Accepts input from multiple monitors.
- Pluggable architecture so that every layer can be replaced to accommodate different solutions.
- Native NoSQL solution.
- Guaranteed high throughput and data synchronization through the BigQueue component.

<table>
<thead>
<tr>
<th>Monitoring Plugin Manager and Scheduler</th>
<th>Monitoring Manager</th>
<th>Data Management Layer</th>
<th>Wide-Column Datastore Driver</th>
<th>Wide-Column Datastore (e.g., HBase or Accumulo)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ceilometer Plugin</td>
<td>Perfmon Plugin</td>
<td>Yammer Plugin</td>
<td>Apache Plugin</td>
<td>MySQL Plugin</td>
</tr>
</tbody>
</table>
BigQueue Component

- An intermediate buffer to synchronize input from multiple monitoring agents.
- The queue is flushed either by size or by time.
- One large request vs many small requests.
- Thread-safe design.
Experiments to evaluate BigQueue

• HBase and Accumulo as the base NoSQL database system.
  • To validate the modularity of the framework.
  • Accumulo is supposed to be a multi-master system, thus closer to the BigQueue concept.

• Comparable VM clusters (4 workers of the same size). Same budget.

• Workload: write requests of 300 to 10000 rows per second with a step of 100 rows.
  • The experiments were repeated three times to account for experimental bias and variability.

• We measured the response time (operational delay) of the database.
HBase results

with default driver

- Linear increase (due to master saturation)
- About 35 s for 10 K rows.
- Fluctuating and high standard deviation
  - Due to master saturation or cloud variability.

with BigQueue

- About 120 ms for 10 K rows.
- 300 times decrease in response time.
- Steadier and lower standard deviation
  - Only due to master saturation.
Accumulo results

with default driver

- Sublinear/constant response time.
- About 1 s in average.
- Fluctuating and high standard deviation.

with BigQueue

- About 200 ms in average.
- 3 to 5 times decrease in response time.
- Fewer spikes (node saturation, cloud variability or experimental variability)
Discussion

• Monitored data is Big Data!
  • Large volume and high throughput
  • Highly multi-dimensional
  • Crucial importance for self-adaptive systems with respect to accuracy and speed of reaction.

• NoSQL datastores offer significant advantages as back-end support for monitoring systems.

• BigQueue proved to significantly improve throughput and response time of even the faster datastores.

• The framework is to be incorporated in an existing autonomic management system and deployed on cloud as a service.

• We plan experiments to evaluate the framework’s contribution in software adaptation overall (also assess read throughput and response time).
Questions?

Thank you!