Towards Combining Online & Offline Management of Big Data Applications

Brian Laub, Chengwei Wang, Karsten Schwan, and Chad Huneycutt

*Georgia Institute of Technology*

{brian.laub, flinter, karsten.schwan, chadh}@cc.gatech.edu
Introduction

• Today’s datacenters process massive data in real-time, for critical apps
  – Business analytics, microsales, spam detection...
• Composed of complex distributed systems
  – Thousands of nodes, many software components
  – Distinct, hard to find failure scenarios
  – Multi-tenant virtualized environments
• Efficient and effective monitoring is key!
Problem Description

• Traditional monitoring systems collect and archive metrics
  – Ex: Ganglia, Nagios, OpenTSDB
  – Historical DB, basic alerting... but view is limited!
  – Operator still must analyze data to pinpoint problems

• Emerging research focuses on online monitoring
  – Ex: Monalytics, VScope
  – Scalable, online algorithms for fast anomaly detection, data aggregation
  – Provide customized monitoring when we need it
  – Resource-constrained
Problem Description

• Each provides valuable set of tools and info
• ...but no link between these two!
• Goal: bridge gap between online & offline monitoring
  – Identify *when* a problem occurs
  – Identify *what type* of problem it is
System Overview

• Background: Build upon previous work VScope* — scalable monitoring for time-sensitive apps using online analytics

• Distributed Processing Graphs (DPGs)
  – Scalable overlay networks deployed dynamically that execute customized monitoring code

• Targeted approach
  – Online algorithms observe short time windows for fast anomaly detection
  – Use DPGs to narrow down problem location quickly, then perform in-depth troubleshooting

• Lacks ability to archive data (history is lost)

* Refer to [17]: “VScope: Middleware for Troubleshooting Time-Sensitive Data Center Applications” by C. Wang, et. al, ACM Middleware 2012.
System Overview

• **VStore** – new extension to VScope that adds offline data collection
  – Built on OpenTSDB/HBase

• VStore API
  – Acts as an OpenTSDB client
  – Hook into DPGs, customized data archiving
  – Provides a comm. layer between customized monitoring and DB backend
System Overview

Diagram showing a structure labeled "Next link in DPG" and a "VStore" connected to it. The diagram includes nodes labeled 1, 2, and 3, and a calculation labeled "avg(1, 2, 3)".
System Overview

• Benefits: VScope + VStore can provide customized monitoring without losing history
• Online algorithms for fast anomaly detection, targeted debugging
• Offline data allows for rich historical analysis
• Capture anomalies that online algorithms might miss
Experiments

• Distributed log-collection workload in a cloud environment (OpenStack+Hadoop+HBase+Flume)

• Experimented with 3 typical scenarios:
  – Periodic process interruption, misconfiguration, network faults
  – Ran VScope to monitor hosts online, using VStore in custom DPG to archive data to OpenTSDB

• Goal: highlight scenarios where VScope’s online monitoring is limited
Periodic Process

- Ex: log archiving, major compaction, MR job
- 2 cases: CPU spike, intense I/O (Hadoop tier)
- Takeaway: online algo. with narrow time window might miss periodic behavior; must watch multiple metrics across tiers
Software Misconfiguration

- Ex: incorrect region splitting, misconfig causing load imbalance
- Takeaway: imbalance is not obvious with online algorithms that aggregate data (e.g. an average)

(a) CPU Utilization at HBase Region Servers

(c) Network traffic at HDFS nodes
Network Faults

- Ex: wrong link speed, VLAN or SDN misconfiguration, “limpware”
- Takeaway: historical data can highlight slowly degrading perf; online analytics must watch app metrics too!

(a) Network traffic at Flume agents

(b) Avg rate of records processed by Flume aggregation agents
Uses for VStore

• Targeted debugging with full history
  – Integrate w/ VScope to offer snapshots of historical data
  – Run baseline “watch” DPG for continuous monitoring, archiving with VStore
  – Operator launches new DPGs to debug specific problems
  – Historical data from VStore shown side-by-side for comparison
Uses for VStore

• Detecting “limplock”*
  – Failing hardware slows system, but doesn’t trigger fail-stop
  – VScope targets monitoring on systems prone to limplock
  – History data from VStore to distinguish from normal load

• Classification of anomalies
  – Use VStore to build database of anomaly scenarios for classification
  – VScope DPGs can run classification algorithms online to aid debugging

Conclusion

- VStore bridges gap between online and offline monitoring
- Historical data helps pinpoint anomalies that online algorithms might miss
- Experiments show promise in using online and offline monitoring for targeted debugging of common fault scenarios
Future Work

• Tight integration w/ VScope
  – Dynamic archiving from multiple points in DPG
  – Automatic retrieval of historical data in VShell
  – Evaluate troubleshooting effectiveness

• Detecting infrastructure faults in clouds (from cloud operator’s perspective)
  – E.g. SDN misconfig causing faults across tenants

• Thorough performance eval
  – Larger scale, real-world IaaS clouds
Questions?

Thanks!