Reducing MTTR and False Escalations: Event Correlation at LinkedIn

Michael Kehoe
Staff Site Reliability Engineer
LinkedIn
False Escalations
Have you ever?

• Been woken because your service is unhealthy because of a dependency?

• Been woken because someone believes your service is responsible?

• Spent hours trying to work out why your service is broken?
Agenda

- Project Problem Statement
- Project Goals
- Architecture Considerations
- Correlation Engine Overview
- Results & Takeaways
- Questions
Michael Kehoe

$ whoami

- Staff Site Reliability Engineer (SRE) @ LinkedIn
- Production-SRE team
- Funny accent = Australian + 3 years American
Michael Kehoe

$ whatis PROD-SRE

- Production-SRE
  - Develop applications to improve MTTD and MTTR
  - Build tools for efficient site issue troubleshooting, issue detection & correlation
  - Provide direction on site monitoring
  - Assist in restoring stability to services during site critical issues
Problem Statement

Service Complexity

- Reliability
- Learning Curve
- MTTR
Problem Statement

Project Technical Goal

Find problem with a service between a given time period (or ongoing) using:

Unified API

Web Frontend
Problem Statement
Project Success Criteria

• Reduce MTTR on incidents
• Reduce false/ needless escalations
Problem Statement

Expected Use-Cases

Applicable use-cases:
• A service has high latency or error rates
  • Find the problematic service(s)

Non-applicable use-cases:
• External monitoring services show slow page-load times
Architecture Considerations

Real-Time metrics analytics (stream processing)

Ad-Hoc metrics Analytics

Alert Correlation
Architecture Considerations

Evaluation

• Real-Time metrics analytics (stream processing)
  • Pros
    • Fast response time
    • Ability to do advanced analytics in real-time
  • Cons
    • Resource intensive (especially at LinkedIn scale)
Architecture Considerations

Evaluation

• Ad-Hoc metric analytics
  • Pros
    • Smaller resource footprint
  • Cons
    • Analysis time is slow
Architecture Considerations

Evaluation

• Alert Correlation
  • Pros
    • Leverage already existing alerts
    • Strong signal-to-noise ratio
  • Cons
    • Analysis constrained to alerts only (boolean state)
Architecture Considerations

Evaluation

• Real-time analytics is expensive, but useful

• Ad-Hoc metric analytics is slower, but cheaper

  • Alert Correlation gives us strong signal
Correlation Engine Overview

At LinkedIn, we had two smaller projects that we could leverage

Drilldown + Site-Stabilizer
  Near-Time metric analytics & event correlation

Invisualize
  Alert Correlation

Existing knowledge available
The ability to correlate is great!

But you need to understand dependencies

Build a callgraph!
LinkedIn applications emit metrics on a per-API and per-dependency basis.

Map metrics to understand dependencies.

Simple to build callgraph platform!
Callgraph

Correlation Engine Overview

Collect:
- Call count
- Latency

Callgraph-be

Voldemort (RO Datastore)

Espresso (RW Datastore)
Correlation Engine Overview

drilldown (Near-Time analytics)

Using callgraph, identifies high-value dependencies (and the associated metrics)

In 5min chunks, analyses high-value metrics
   Using a k-means unsupervised algorithm, find similar trends between service metrics

Queryable API

Outputs correlation confidence scores
   Normalised between 0-100

<table>
<thead>
<tr>
<th>Service</th>
<th>Confidence score</th>
</tr>
</thead>
<tbody>
<tr>
<td>cap-backend</td>
<td>79.8541782917</td>
</tr>
<tr>
<td>orms-backend</td>
<td>14.832181796</td>
</tr>
<tr>
<td>fuse-server</td>
<td>8.75344187723</td>
</tr>
</tbody>
</table>
Correlation Engine Overview

inVisualize (Alert Correlation)

inVisualize analyses alerts (in realtime) from each service

Use callgraph to calculate the unhealthy service and affected services

Queryable API
- Results normalised between 0-100

Visualizes impact
Correlation Engine Overview

inVisualize
Correlation Engine Overview

Site-Stabilizer

Backend service
  Collates recommendations from Drilldown & inVisualize

Decorates recommendations with:
  Scheduled changes
  Deployment events
  A/B experiment changes
Correlation Engine Overview

Architecture

- Callgraph-api
  - Callgraph-be
    - Invisualize
    - Site-stabilizer
    - Drilldown
Correlation Engine Overview
Correlate-fe

API for automation
  Auto-remediation
  Alert suppressing

UI for manual introspection
Correlation Engine Overview

Correlate-fe

User Interfaces gives
Responsible service
Correlation Confidence
Root cause
SRE team
Analysis

<table>
<thead>
<tr>
<th>Correlation API Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Responsible Service</td>
</tr>
<tr>
<td>Correlation Confidence</td>
</tr>
<tr>
<td>Root Cause</td>
</tr>
<tr>
<td>Responsible Service Oncall</td>
</tr>
<tr>
<td>Analysis</td>
</tr>
<tr>
<td></td>
</tr>
</tbody>
</table>
Correlation Engine Overview

Architecture

correlate-fe

site-stabilizer

drilldown

invisualize

Callgraph-api

Callgraph-be
Latency Alert

Nurse Plan arguments
- service-name: my-frontend
- req_confidence = 85
- escalate = True

Find what's wrong with 'my-frontend' in DatacenterB

Service: Service-C
Confidence: 91%
Reason: 'Service-C' has high latency after a deploy
Service Owner: SRE

Alert Correlation API

Escalate to correct SRE

Iris
Early Results

Siteops (NOC) has greater visibility on the site

Reducing MTTR

Reducing false escalations
Conclusion

Understand what correlation approach makes sense for you

Understand your dependencies

Build, Integrate and benefit!
Team

Michael Kehoe
Rusty Wickell
Reynold Perumpilly
Govindaluri Kishore
Renjith Rajan
Questions?
Correlation Engine Overview

Call count

Latency

Callgraph

Callgraph-be

RestLi (Internal API's)

Voldemort (RO Datastore)

Espresso (RW Datastore)
Correlation Engine Overview

Architecture

correlate-fe

site-stabilizer

drilldown

invisualize

Callgraph-api

Callgraph-be