Automated Behavior Detection
(for Utility Clouds)

Karsten Schwan (schwan@cc.gatech.edu)

Vanish Talwar (HP Labs)
Partha Ranganathan (HP Labs)
Current Monalytics Architecture

Coordinators hierarchy at Node N

Monitoring functions

Management functions

Virtualization Layer

Sensor/Actuator metadata registration
Sensor Delivery Control/DataPath
Bootup Overlay-Discovery Query-Functions

Hardware Layer

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator

Sensor Actuator
Monalytics - Scalability

**Dynamics I:**
Local analysis and filtering

<table>
<thead>
<tr>
<th></th>
<th>Request Trace Records Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>Without Local Analysis</td>
<td>1.41. MB</td>
</tr>
<tr>
<td>With Local Analysis (via filtering)</td>
<td>60.45 KB</td>
</tr>
</tbody>
</table>

**Dynamics II:**
Zoom-in Analysis

<table>
<thead>
<tr>
<th></th>
<th>Monalytics Run(3hr)</th>
<th>Offline Analysis(10hr)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Centralized: Data Transferred</td>
<td>394.08 KB</td>
<td>1.15 MB</td>
</tr>
<tr>
<td>Local Analysis: Data Transferred</td>
<td>123.32 KB</td>
<td>345.6 KB</td>
</tr>
</tbody>
</table>

RUBiS Testcase – details below
Monalytics – Behavior Detection

**Aggregation Problem**
1. **Scalability:** reduce the data volume in communication and analysis
2. **Retain valuable information** for anomaly detection and identification.
3. ‘horizontal crossing' and 'vertical crossing‘: metrics in different levels/components are collectively considered

**Detection Problem**
1. **Online** designating when the utility cloud is experiencing anomalies.
   2. **High Detection Rate and Low False Alarm Rate**
3. Unsupervised method with **minimal pre-knowledge** about normal or abnormal behaviors.

**Zoom-In Problem**
Localizing anomalies so as to narrow the search scopes for further diagnosing the causes of those anomalies.
EbAT: Entropy based Anomaly Testing

or

Scalable Online Anomaly Detection Using Metric Distributions

1. A lightweight online approach to scalable monitoring, capable of raising alarms and zooming in on potential problem area.
2. Detects changes in aggregated metric distributions rather than individual metric values (like threshold-based approaches)
3. Unsupervised, minimal prior knowledge


EbAT Overview

Entropy Timeseries Construction
- normalizing
- data binning
- leaf level?
  - Y: entropy aggregation
  - N: m-event generation
- entropy calculation
- time series

Entropy Timeseries Processing
- Spike Detection:
  1. Visual Identification
  2. Exponential Weighted Moving Average (EWMA)
- Signal Processing:
  Wavelet Analysis
- Subspace Analysis:
  Online Singular Value Decomposition (SVD)

Metrics Collection
- OS metrics
- app. metrics
- platform metrics
- core level metrics
- underlying entropy timeseries

Aggregating metric distributions
# EbAT Results – RUBiS Failure Injection

<table>
<thead>
<tr>
<th>Methods</th>
<th>Description</th>
<th># Alarms</th>
<th># Successful Detections</th>
<th>Recall</th>
<th>Precision</th>
<th>Accuracy((F_1))</th>
<th>FAR</th>
</tr>
</thead>
<tbody>
<tr>
<td>Entropy I</td>
<td>Global Entropy Using Entropy of Child Entropies</td>
<td>45</td>
<td>43</td>
<td>0.86</td>
<td>0.96</td>
<td>0.91</td>
<td>0.04</td>
</tr>
<tr>
<td>Entropy II</td>
<td>Global Entropy Using Sum of Child Entropies</td>
<td>56</td>
<td>45</td>
<td>0.90</td>
<td>0.80</td>
<td>0.85</td>
<td>0.20</td>
</tr>
<tr>
<td>Threshold I</td>
<td>Near-Optimum</td>
<td>46</td>
<td>33</td>
<td>0.66</td>
<td>0.72</td>
<td>0.69</td>
<td>0.28</td>
</tr>
<tr>
<td>Threshold II</td>
<td>Static &gt;0.9 or &lt;0.05</td>
<td>18</td>
<td>16</td>
<td>0.32</td>
<td>0.89</td>
<td>0.47</td>
<td>0.11</td>
</tr>
</tbody>
</table>

\[
\text{Recall} = \frac{\text{# of successful detections}}{\text{# of total anomalies}}
\]

\[
\text{Precision} = \frac{\text{# of successful detections}}{\text{# of total alarms}}
\]

\[
\text{Accuracy}(F_1) = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}
\]

\[
\text{False Alarm Rate (FAR)} = \frac{\text{# of false alarms}}{\text{# of total alarms}}
\]

**On Average 57.4% Improvement on Accuracy, 59.3% Reduction on FAR**

**Additional Results: Hadoop 80 node OpenCirrus runs + add. metrics**
Many Remaining Issues

• Scaling to Exascale:
  – dynamics: needed: scalable control, including:
    – automation in deployment and use (e.g., monalytics QoS)
  – ease of use: higher level abstractions, including
    – linking abnormal behavior detection to problem diagnosis and prevention
  – real-life systems: distributed utility clouds
Conclusions and Future Work

• Monitoring for effective management: Monalytics
  – rich domain for future work – software architectures, systems and platform support, methods and techniques
  – close linkage with business analytics and real-time data analysis (for us: linkage to HPC)
  – encouraging cooperation and joint research: no one system does it all, need for many methods and structures, many unsolved problems, …

• From Monalytics to Large-scale Management:
  – managing over time (simple example in vManage) to react to system and application changes over time (little understood)

• Dynamic change in management purpose and different/multiple simultaneous management goals

• Can we make systems more manageable?
  – constructing more robust (performance robustness and reliability) virtualization infrastructures – runtime learning?
  – platform support for monitoring, management, and coordination