Real Time Analysis and Event Prediction Engine

Joshi Fullop, Ana Gainaru, Joel Plutchak
Terminology

- **Event**
  - The description of what has happened.

- **Occurrence**
  - The instance of an event that takes place at a given time and location.

- syslog provides time and location with a string that describes the occurrence, but not necessarily the event.
Example: Event manifestations

- **Event**: CPU temperature has exceeded a threshold of 85°C
- **Manifestations in a log message**:
  - 20120428 12:00:01, node001, “CPU temperature critical: 91.3°C”
  - 20120428 12:00:02, node002, “CPU temperature critical: 90.5°C”
  - 20120428 12:00:03, node003, “CPU temperature critical: 90.5°C”

- Q: How do you get from syslog messages to an event?
- A: Regular expression engine, right?
Problems with regular expressions

- Requires prior knowledge and domain expertise
- Only identifies things specifically coded for
  - Everything else slips through the cracks
- Regular expression might not be coded perfectly
- What if software changes or gets updated?
  - Before update: “CPU temperature critical: 91.3C”
  - After update: “CPU temperature is critical: 196.3F”
Hierarchical Event Log Organizer (HELO)

- Intelligent, learning log stream identifier
- Analyzes logs to build a template library
- Identifies & tags log messages with a template ID
  - If a log message does not match a template
    - Existing similar template is modified slightly so that the new message and old messages match
    - Or a new template is created
- Prepares log messages for entry into a database
- Provides derivative data. E.g. rate of new events
HELO additions

- Parallelization
  - Distributed log identifiers w/central template library
- Source tagging
  - Handles logs from various subsystems
- Template grouping
  - Can group templates into statistically similar sets
  - i.e. those containing ‘eth#’ get grouped and mapped into a networking supergroup
Event Correlation Analysis

• All-to-all mining process
  • For each event (B), we find for each other event (A), the time that A precedes B and add it into a set.
  • Perform a DBScan cluster algorithm on that set to find clusters of data.
  • Record each cluster’s
    • average time
    • standard deviation
    • probability
    • count
Correlated Events Map
Event chain prediction calculations

If Event A occurs at 1335675600 unixtime.
Probability of Z = 50% * 75% = 37.5%
Time until Z = (10+5)*60 (seconds) = 900 seconds from A (@1335676500)
Window for Z = (2*1 + 2*1) * 60(seconds) = 240 seconds

We can predict that there is a 37.5% chance that Z will occur at 1335676500 with a window of +/- 240 seconds.
Graphs are sets of chains

Directed Graph

A chain within a Graph
Prediction algorithm

• Select Events of Interest (EoI)
• For each EoI, backwards traverse graph, looking for the occurrence of its preceding event within the time window (avg time + 2 * std.dev) from now
• If not found, recursively traverse backwards, adding previous path’s time window until
  • A) an occurrence is found
  • B) the chain terminates
  • C) you reach a boundary condition of maximum time window, number of steps from EoI, or time spent computing
Summary

• We rarely ever know what to look for going into the building of a new supercomputer.
• So we built a system to figure out how things correlate.
• Then we found that we can use those relationships to predict future event occurrences.
Contact info

• Joseph ‘Joshi’ Fullop
  • jfullop@ncsa.illinois.edu