Machine Learning Conference

March 6-7 2014

EVENT STREAM DIAGNOSTICS

Yogesh Padmanaban, Office JETS Team

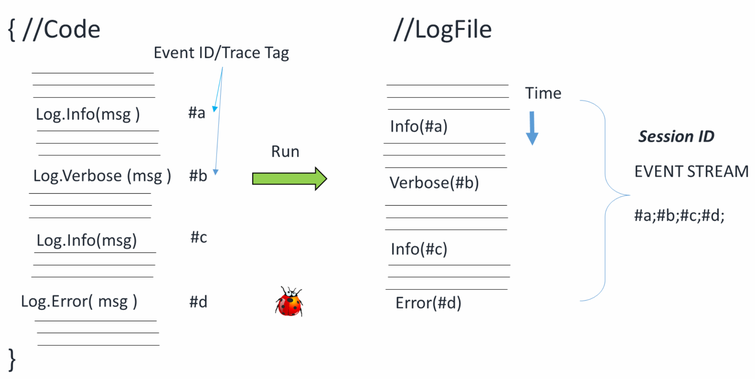
Motivation

In the services world, a significant amount of time and effort is spent investigating live site issues. A critical portion of root cause analysis involves determining the pathway of code execution which led to the issue. This can be improved.

Approach

1. Find all the observed pathways of execution from logs (Event Streams).
2. Identify the Event Streams leading to errors.
3. Characterize expected service behavior from pathway trends and identify anomalies.
4. Determine if similar pathways lead to similar bugs.
5. Explore potential error scenarios from analyzing multithreaded Event Streams.

Visualizing the Pathways of Code Execution: Event Stream



I. Experiments on Office Client Services

- **Nexus** – a pipeline for uploading Office client usage data to COSMOS. 3.5 Gb of logs has 96 Event Streams from 810k client requests.
- **Roaming** – a system for sharing user-specific information across devices. 3.5 Gb of logs has 353 Event Streams from 110k requests.

Data Source: 3 days worth of Unified Logging Service (ULS) log files.

II. Most Erred Event Streams

NEXUS EVENTSTREAMS	FREQUENCY	SUM OF ERRORS	STREAM LENGTH
apl4f;arjvu;an3j0;an3j7;an3j8; apl4i;aopkg;**aopkn;aopkj;aojak**	**912**	**2736**	**10**
an8jj;aq41j;**an8jm;an8jm;**ahv0v	**489**	**978**	**5**
an8jj;aq41j;**an8jm;an8jm;an8jm; an8jm;**ahv0v	**14**	**56**	**7**

Mapping Source File and Components to the Event Stream

Event Stream:

apl4f;arjvu;an3j0;an3j7;an3j8;apl4i;ap3m2;ahv0v;aopkg;aopkn;aopkj;aojak

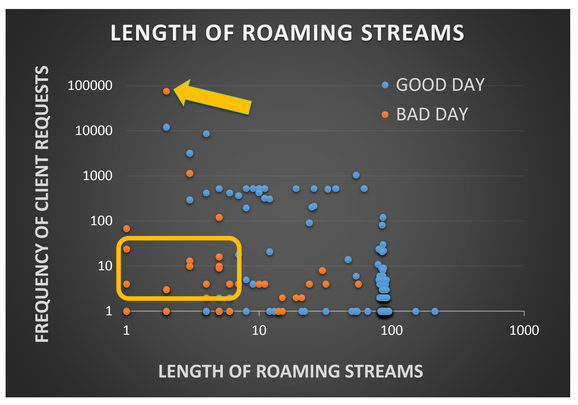
EVENTID	FILE PATH	COMPONENT
ap4lf	nexus\loggingmessagehandler.cs	NEXUS
arjvu	nexus\reverseipmapper.cs;	NEXUS
an3j0	nexus\uploadcontroller.cs	NEXUS
• • •	• • •	• • •
aojak	nexus\payloadprocessorworker.cs;	NEXUS

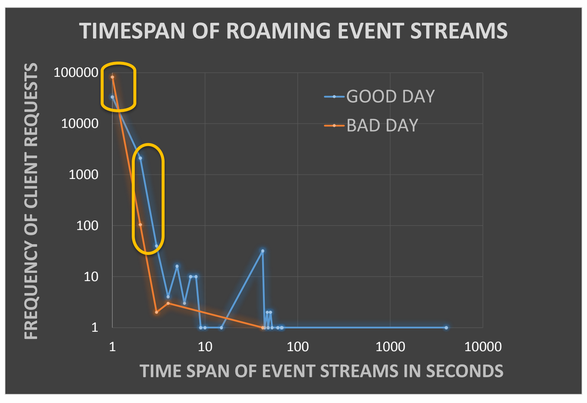
Path Stream:

nexus\loggingmessagehandler.cs; nexus\reverseipmapper.cs; nexus\uploadcontroller.cs(3); nexus\loggingmessagehandler.cs;nexus\batchedtelemetrystoragewriter.cs; msoserviceplatform\periodictimer.cs;nexus\ulsruleresultreader.cs(3)**[2]**; nexus\payloadprocessorworker.cs;

Component Stream: NEXUS(7); MSOSERVICEPLATFORM (1); NEXUS(4)**[3]**;

III. Detect Anomalies in Service behavior using Length and Timespan trends of Event Streams





IV. Similar Errors & Similar Event Streams

Could similar pathways lead to similar bugs?

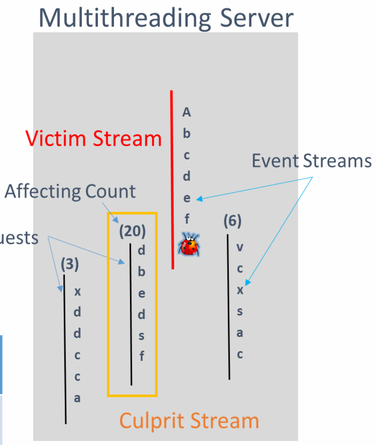
We want to identify similar Event Streams in each service and study their behavior. We consider Event Stream as a "**string**" of event IDs with each event ID as a "**character**" to apply the following string metrics.

- Edit Distance or Levenshtein Distance
- Longest Common Subsequence Length
- BLAST score using Smith Waterman Algorithm
- Sorenson Dice's coefficient or the Bigram score.

NEXUS EVENT STREAMS	FREQUENCY	MEDIUM	ERRORS
apl4f;arjvu;an3j0;an3j7;an3j8;apl4i;aopkg; aopkn;aopkj;aojak;ap3m2;ahv0v	3	27	9
apl4f;arjvu;an3j0;an3j7;an3j8;apl4i;aopkg; apkh5;apkh8	141	1269	0
apl4f;arjvu;an3j0;an3j7;an3j8;apl4i;aopkg; aopkn;aopkj;aojak;	912	6384	2736

V. Multithreaded Event Stream Interaction

We want to explore scenarios which may lead to errors in multithreaded execution of client requests in Roaming Service.



VICTIM STREAM	CULPRIT STREAM	AFFECTING COUNT
7a3;aebci;af7xc;aiwcp;akhz0; adhof;adhoi;af7xd;77a3	77a3;aebci;af7xc;aiwcp;akhz0; adhof;adhoi;af7xd;77a3	**12**
77a3;aeqvg;aeqvs;adhoi;ajkyf; aeqvu;aeqvv;ajmah;aeqvv; ajmah;aeqvy;77a3	77a3;77a3	**11**

Microsoft

ENGINEERING EXCELLENCE
A Trustworthy Computing Group

Supporting your pursuit of excellence.

Paper #24: Event Stream Diagnostics