The University of Texas at Austin

Software Testing

Software Engineering - Option III

Testing Distributed Systems - Kafka

Authors:
Howie Benefiel
Dale Pedinski

Professor:
Dr. Sarfraz Khurshid

August 14, 2018

Abstract

1 Introduction

2 Kafka Architecture

3 Prior Work

From our literature review, we found that the field of testing distributed systems is relatively unexplored. This is backed up by the findings of Yuan et al. [**?**].

The authors of Yuan et al. analyzed a number of open-source distributed systems running production workloads in industry currently. Their findings showed that a large percentage of catastrophic failures could have been prevented by simple tests and static analysis.

In Yuan et al.ś paper, they studied the catastrophuc failures 5 distributed, data-intensive systems, Cassandra, HBase, Hadoop Distrbibuted File System (HDFS), Hadoop MapReduce, and Redis. They defined a catastrophic failure as a failure which would result in unavailability of the system or data loss. To determine the faults of the system, the authors analyzed failures reported on the respective system's bug tracker. For each system, the authors analyzed the catastrophic failures.

In general, Yuan et al. found that the failure modes of distributed systems were complex. The authors determined that the complexity of these failure modes was caused by the relatively large state space of distributed systems. Quantitatively, they found that 77% of failures required more thatn one event to produce the failure, and 90% required no more than three.

The authors of Yuan et al. then analyzed the types of inputs required to produce the failure. The most unexpected finding from this analysis is that 24% of failures were caused by a node becoming unreachable. This is surprising because a core tenant of distributed systems is fault-tolerance, i.e., the system should be able to handle node*s* becoming unavailable.

The authors also analyzed the determinism of failures. In distributed systems, there is the possibility that a sequence of events could produce a failure sometimes while not producing a failure other times. The authors found that

74% of failures are deterministic. This is a promising result because it means the failures are not by determined by the timing of the input events.

By far the most unexpceted result of this paper

4 Duck Tape

5 Custom Testing Apparatus

References

A Github Code

https://github.com/howinator/software-testing-kafka-project