**Problem: Real Distributed Systems in Clouds Are Bug-Prone**

- e.g. Cassandra, Flume, HBase, HDFS, MapReduce and ZooKeeper
  - 3 bug tickets per day on average [Gunawi et al. SoCC’14]
  - Almost half of them takes over 1 month to debug
  → Cloud-computing business faces risk of unavailability and data corruption!

**Our main scope of interest:** distributed race condition, fault tolerance bug
(Such bugs are especially peculiar to distributed systems)

**Solution: Model Checker for Unmodified Distributed Systems**

Explores implicit state space by reordering several events to find bugs

**Related Work**

- Not impl.-level model checker
- Not applicable to large software; Java-only
- Requires src-level knowledge; Java-only
- Win32 API only; paper only
- dynlinked POSIX/MPI API only

**Novelty of Earthquake:**
✔ Just Requires Protocol-Level Knowledge*
✔ Applicable to Real Impl. in Any Language*
✔ Supports User-Written Heuristic Plug-In
✔ Open Source (Apache License 2.0)
* OpenFlow/netfilter packet events

**Epic Win: Found Distributed Race Condition of Apache ZooKeeper**

- Bug: “Observer” node cannot be promoted to “Participant”
  - The bug was marked **critical** by ZK community.
  - We sent a bug-fix to ZK community and the fix was merged. [issues.apache.org/jira/browse/ZOOKEEPER-2212](https://issues.apache.org/jira/browse/ZOOKEEPER-2212)
  - Without Earthquake, the bug could not be reproduced in 5,000 experiments. (about 60 hours)

Earthquake is Available at Docker Hub!

$ docker run -i -t osrg/earthquake
Earthquake: An Open-Source Framework of Implementation-Level Distributed System Model Checkers

Akihiro Suda, Hitoshi Mitake, and Tomonori Fujita
NTT Software Innovation Center
lastname.firstname@lab.ntt.co.jp

1. Introduction
Real implementation of distributed systems in clouds are bug-prone. Even matured and well unit-tested softwares (e.g. Cassandra, Flume, HBase, HDFS, MapReduce and ZooKeeper [1]), many bugs of them are being reported every day, and most of bugs need long time to get resolved. Such a bug exposes cloud-computing business to risk of unavailability and data corruption.

Some existing studies (e.g. [2]) showed that distributed system model checkers (DMCKs) are effective to find such implementation-level bugs, especially ones related to message ordering and fault-tolerance. DMCKs explore state space by permuting messages and injected fault events in several orders so as to find bugs. However, applying a DMCK to a real system is difficult, because DMCK requires plenty of implementation-specific, source code-level knowledge.

In this poster, we propose a new open-source DMCK named Earthquake. Earthquake is easy to use because it does not require source code-level knowledge, but only requires protocol-level knowledge to find bugs. If a user has source code-level knowledge, he/she can still make use of it for deeper state exploration.

Orchestrator (Core part of Earthquake): Receives several kind of events from Inspector, permutes them, and send back to Inspectors. By default, Earthquake permutes events in random order. A user can write his/her own permutation heuristic plug-in to alleviate state explosion (as in SAMC [2]) and find bugs efficiently. He/she can also inject fault events (e.g., network partition, node crash and reboot) so as to test fault-tolerance of the target system.

Orchestrator also executes a workload script to run experiments, and a health check script to check whether the target system is hitting bugs.

Inspector: Inspects the target system and send events to Orchestrator. Currently, we have two implementations of Inspectors:

Ethernet Inspector: Inspects Ethernet packets and blocks them until Orchestrator allows to pass. A user is required to write his/her own Inspector to parse semantic information of the packets. Note that he/she does not need source-code level knowledge, but just needs protocol-level knowledge to write an Inspector. Furthermore, Ethernet Inspector is applicable to programs written in any language.

Ethernet Inspector is implemented as a Ryu SDN [3] application. We also provide Linux Netfilter-based implementation for a case where Ryu cannot be installed.

Java Inspector: Inspects Java function calls and blocks them as in packets in Ethernet Inspector. A user has to decide which functions to be inspected. This requires enormous source code-level knowledge as in existing works, but enables much more exhaustive state exploration than

2. Architecture
A typical configuration of Earthquake is shown in the other page.

3. Evaluation
We applied Earthquake to Apache ZooKeeper using Ethernet Inspector without any modification to ZooKeeper. Earthquake successfully found a distributed race condition bug ZOOKEEPER-2212 that had been previously unknown.

ZooKeeper was unintentionally dependent on a specific ordering of ZAB (ZooKeeper Atomic Broadcast) packets and FLE (Fast Leader Election) packets. When an observer in a ZooKeeper ensemble receives a specific kind of ZAB packet after receiving a specific kind of FLE packet, the observer stays at a weird state and cannot be promoted to a participant. Although Earthquake does not fully control non-determinism, Earthquake can easily reproduce this bug in a few experiments. Without Earthquake, we were not able to reproduce the bug in 5,000 experiments. (About 60 hours)

4. Discussion
A major challenge still left is formulation of workloads.

We consider a good workload is the one that is unreliable, even though expected to happen in real business. For example, dynamic reconfiguration [5] is attractive in business, as it makes a cloud system tolerable to seasonal traffic spikes. However, real implementations of dynamic reconfiguration tend to be bug-prone due to complex state transitions. Actually, we found ZOOKEEPER-2212 on the way of testing reconfiguration.

We are also groping for other good workloads.

5. Conclusion
Earthquake is a powerful, open-source DMCK framework for finding implementation-level bugs of distributed systems.

Earthquake is available for download under Apache License 2.0 at http://osrg.github.io/earthquake/. A tutorial for reproduction of ZOOKEEPER-2212 is also included in this repository.

References