Acto: Automatic End-to-End Testing for Operation Correctness of Cloud System Management

Jiawei Tyler Gu Xudong Sun Wentao Zhang Yuxuan Jiang
Chen Wang Mandana Vaziri Owolabi Legunsen Tianyin Xu

University of Illinois Urbana-Champaign · IBM Research · Cornell University

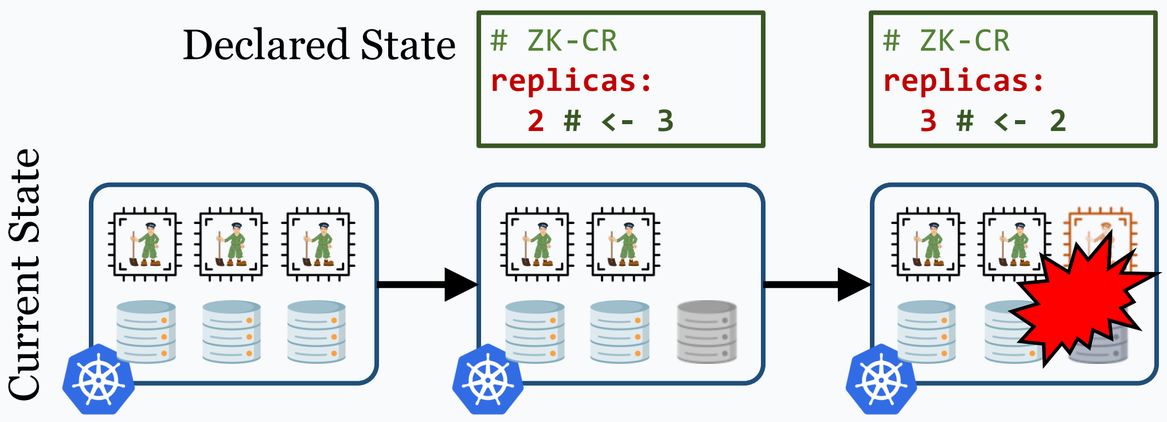
https://github.com/xlab-uiuc/acto

0. Contributions

- Acto: **automatic end-to-end testing** for cloud-system operators
  - Checking **operation correctness**:
    - *Always* reconciling the managed system to the desired state
    - *Always* recovering the system from undesired or error states
    - *Always* being resilient to operation errors
- A **"push-button"** tool for unmodified Kubernetes operators
  - Covering every interface property *at least once*
- Acto has detected **56** bugs (**42** confirmed and **30** fixed) in **11** popular Kubernetes operators

1. Background & Motivation

- Modern cloud systems are managed by **operators**
  - Implementing **declarative interface**
- Operator correctness is critical to system reliability

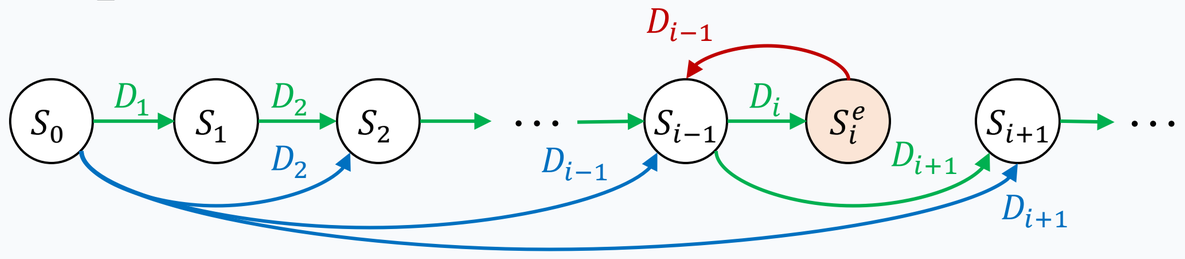


A bug detected by Acto in the Pravega's ZooKeeper operator

- Testing for operation correctness is challenging
  - Exploring different transitions of system states
  - Generating effective state declarations
  - Automatically checking validity of system states

2. How Acto Explores State Transitions

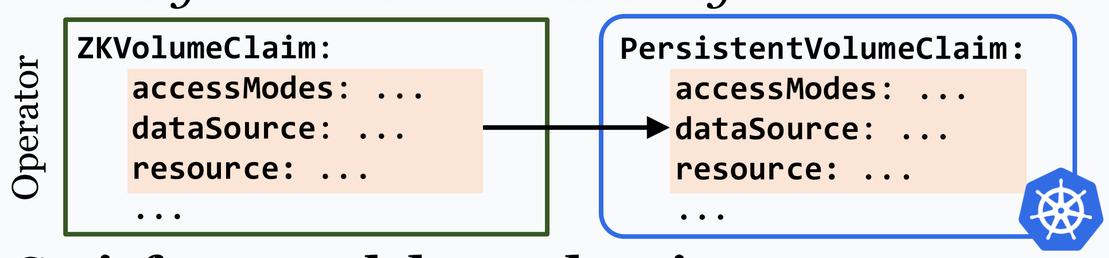
- Modeling an operation as a pair of the current state S^c and the desired state D
- **Single operation:** testing if the operator can reconcile the system to the same state from different S^c given the same D
- **Operation sequence:** testing operations from different non-initial starting states
- **Error-state recovery:** testing if the operator can restore the managed system from **implicit** or **explicit** error states



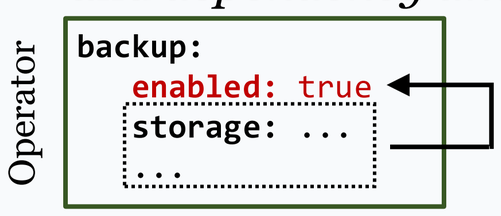
A test campaign that combines all three test strategies

3. Generating State Declarations

- **Satisfy value constraints** (from OpenAPISchema)
- **Exercise representative scenarios**
  - Inferring property semantics through *structure analysis* and *static taint analysis*



- **Satisfy control dependencies**
  - Inferring dependencies via *naming convention* and *dependency analysis*

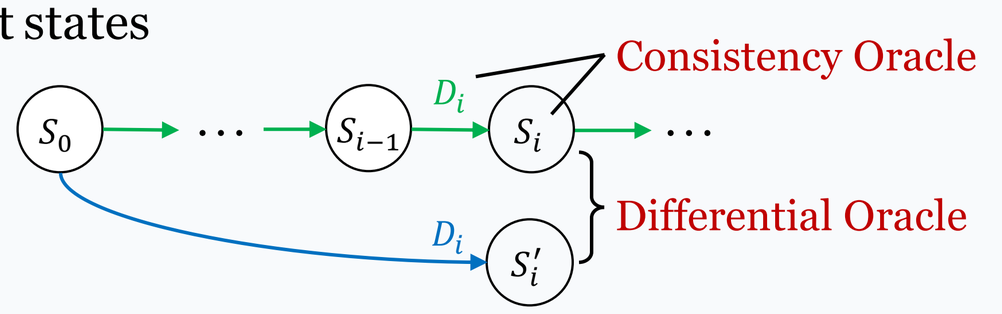


Control dependency

`storage` only takes effect when `enabled` is `true` (i.e., `backup` is enabled).

4. Automatic Test Oracles

- Checking **explicit** error states (e.g. exceptions, panic)
- Automatically detecting **implicit** state mismatches
  - **Consistency oracle** checks if the operator view is **consistent** with the Kubernetes view
  - **Differential oracle** checks if the system is reconciled to the **same** end state from different start states



5. Evaluation

- Can Acto **effectively** find new bugs?
  - Acto has found **56** bugs in **11** operators
- Does Acto find bugs **efficiently**?
  - Acto tested each operator with a **nightly** run
- Are Acto's testing results **trustworthy**?
  - Acto-□: no FP; Acto-■: a **0.19%** FP rate

Operator	Undesired State	System Error	Operator Error	Recovery Failure	Total
CassOp	2	0	0	2	4
CockroachOp	3	0	2	0	5
KnativeOp	1	0	2	0	3
OCK/RedisOp	4	0	3	1	8
OFC/MongoOp	3	1	2	2	8
PCN/MongoOp	4	0	0	1	5
RabbitMQOp	3	0	0	0	3
SAH/RedisOp	2	1	0	1	4
TiDBOp	2	1	0	1	4
XtraDBOp	4	0	1	1	6
ZooKeeperOp	4	1	0	1	6
Total	**32**	**4**	**10**	**10**	**56**