ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE

ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL

ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

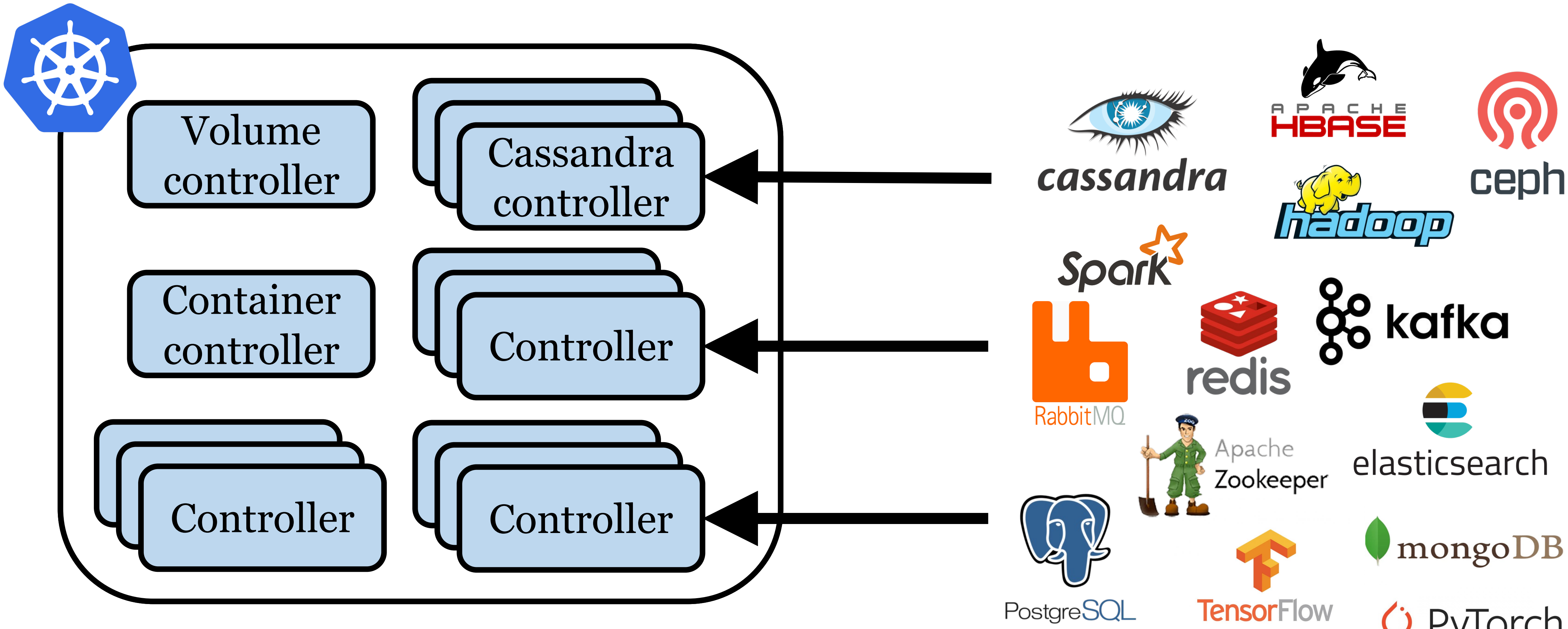
Automatic Reliability Testing for Cluster Management Controllers

Xudong Sun, Wenqing Luo, Jiawei Tyler Gu, Aishwarya Ganesan, Ramnatthan Alagappan, Michael Gasch, Lalith Suresh, Tianyin Xu

UNIVERSITY OF ILLINOIS URBANA-CHAMPAIGN

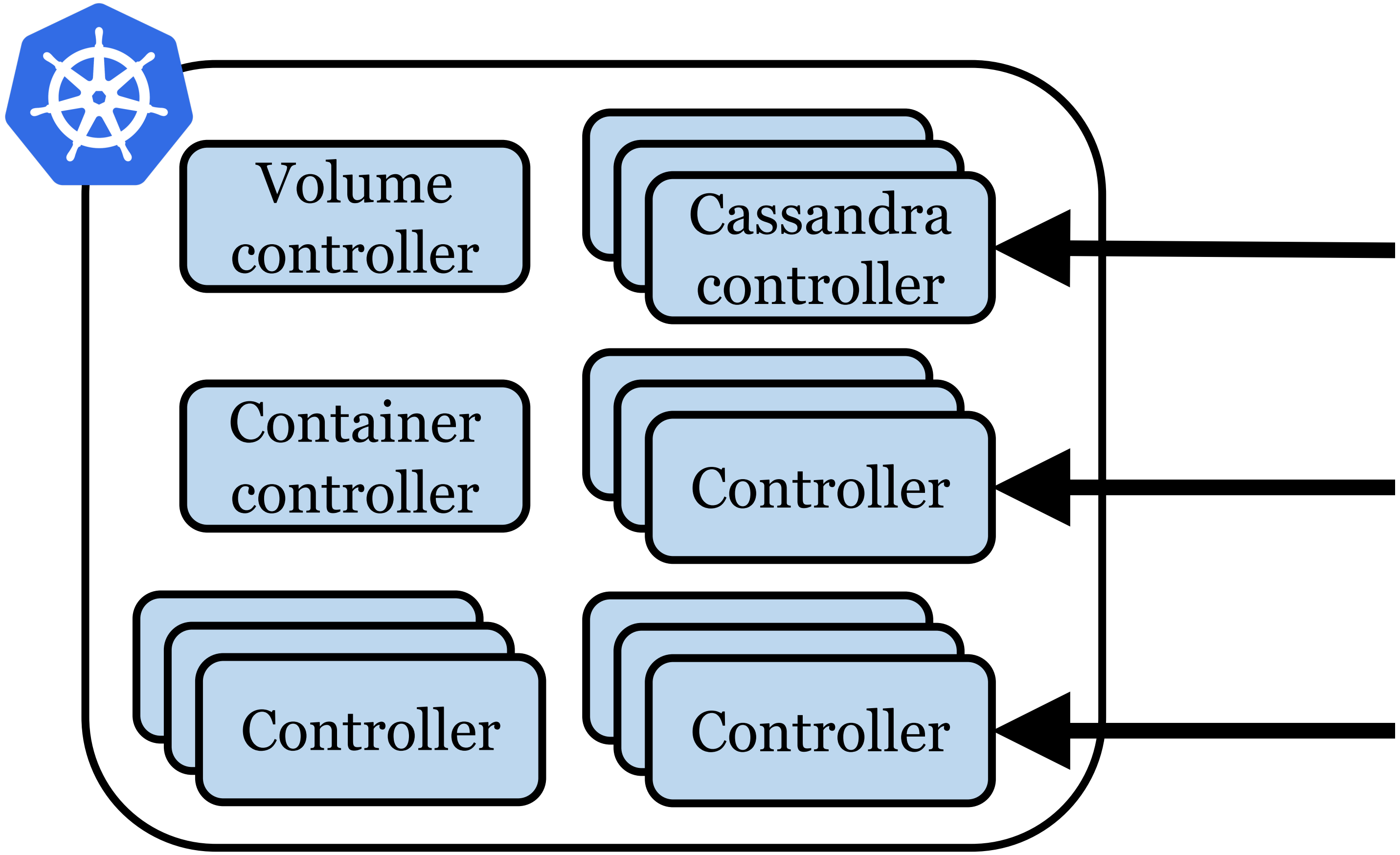
vmware

Cluster management is realized by controllers



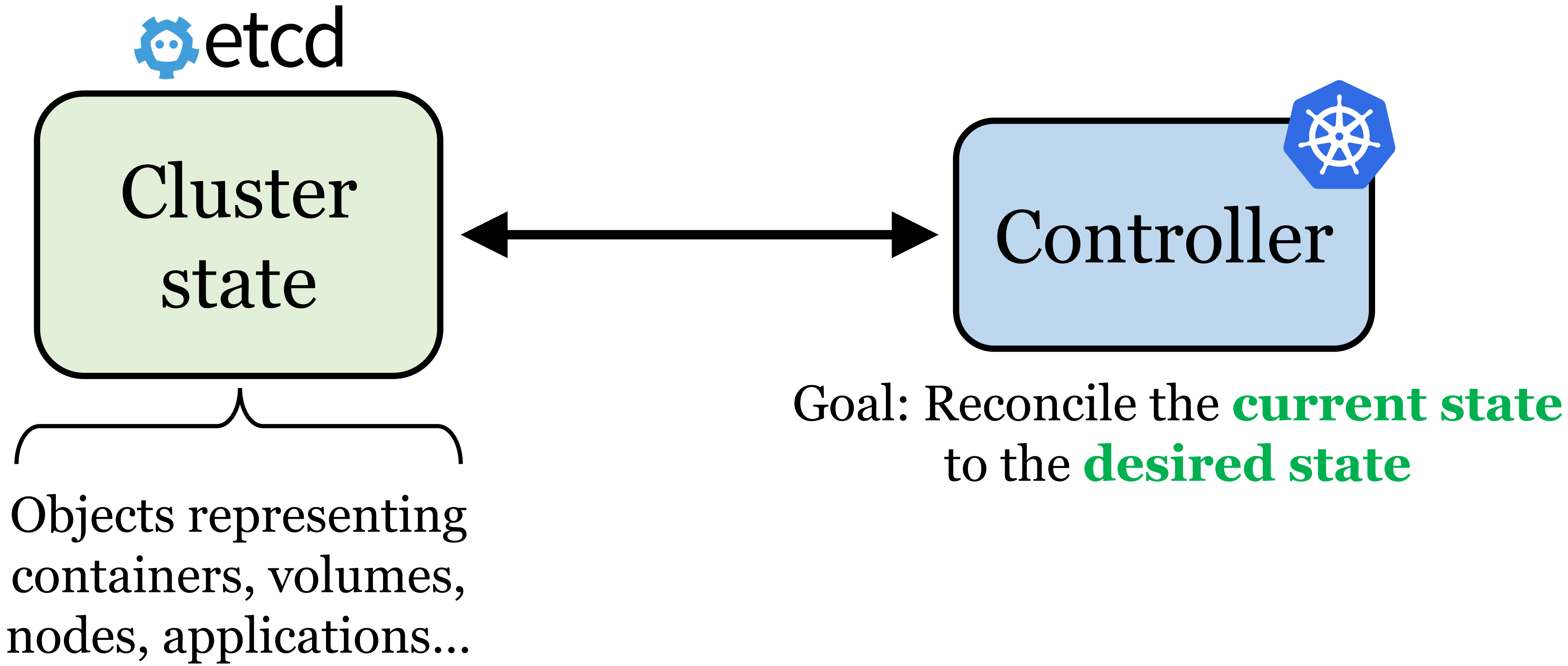
Thousands of controllers being developed
by the Kubernetes community

Cluster management is realized by controllers

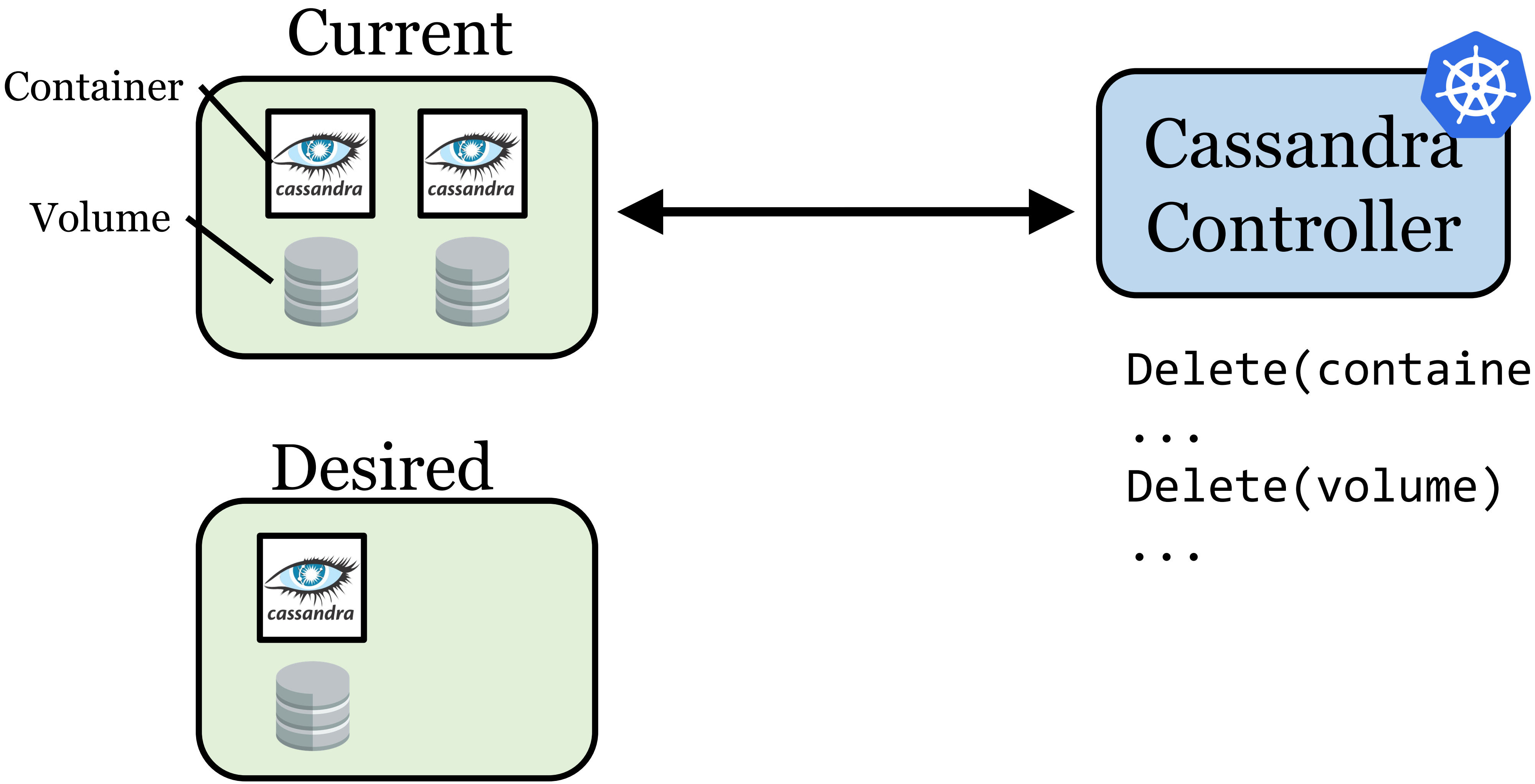


Thousands of controllers being developed by the Kubernetes community

Controllers implement state reconciliation

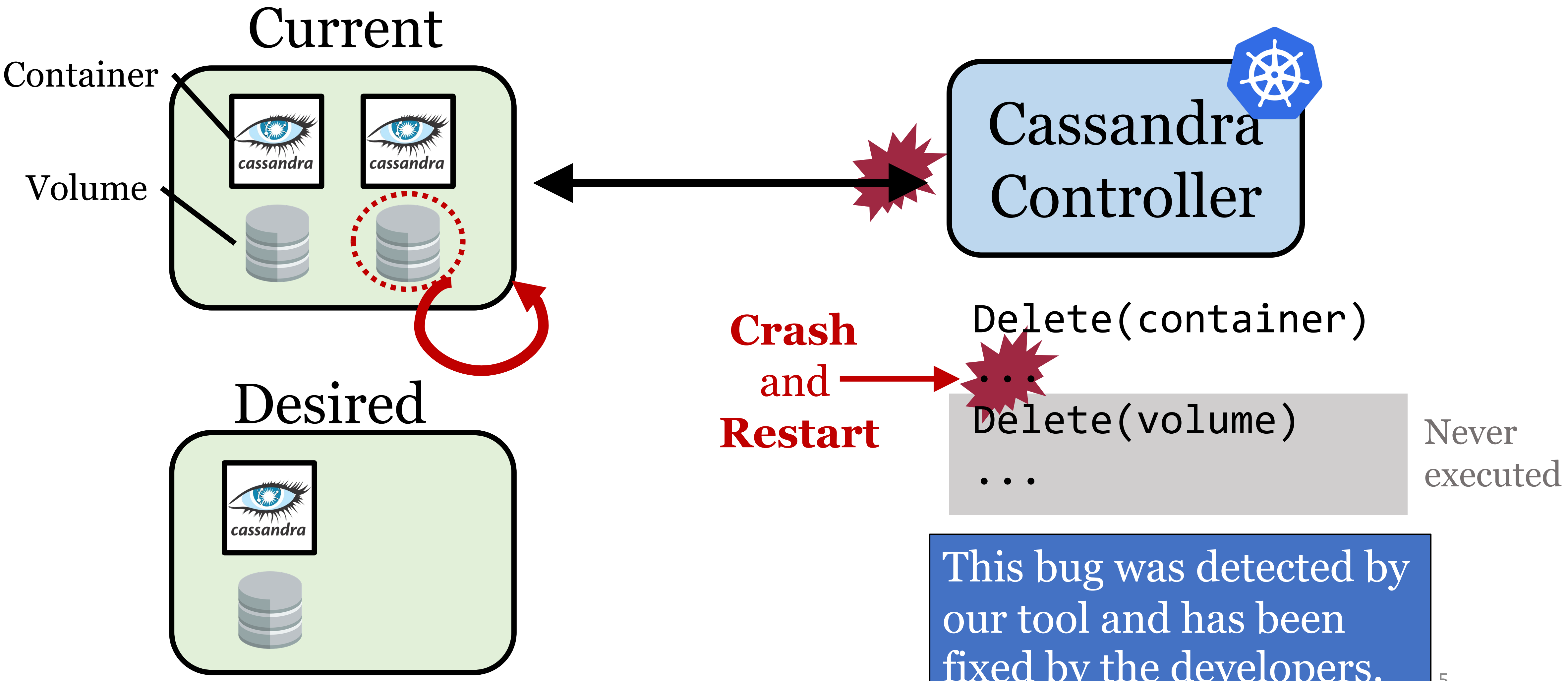


Controllers implement state reconciliation




```
Delete(container)
...
Delete(volume)
...
```

Controller reliability is critical, but challenging!



This bug was detected by our tool and has been fixed by the developers.

Contributions

ARTIFACT EVALUATED usenix ASSOCIATION AVAILABLE
ARTIFACT EVALUATED usenix ASSOCIATION FUNCTIONAL
ARTIFACT EVALUATED usenix ASSOCIATION REPRODUCED

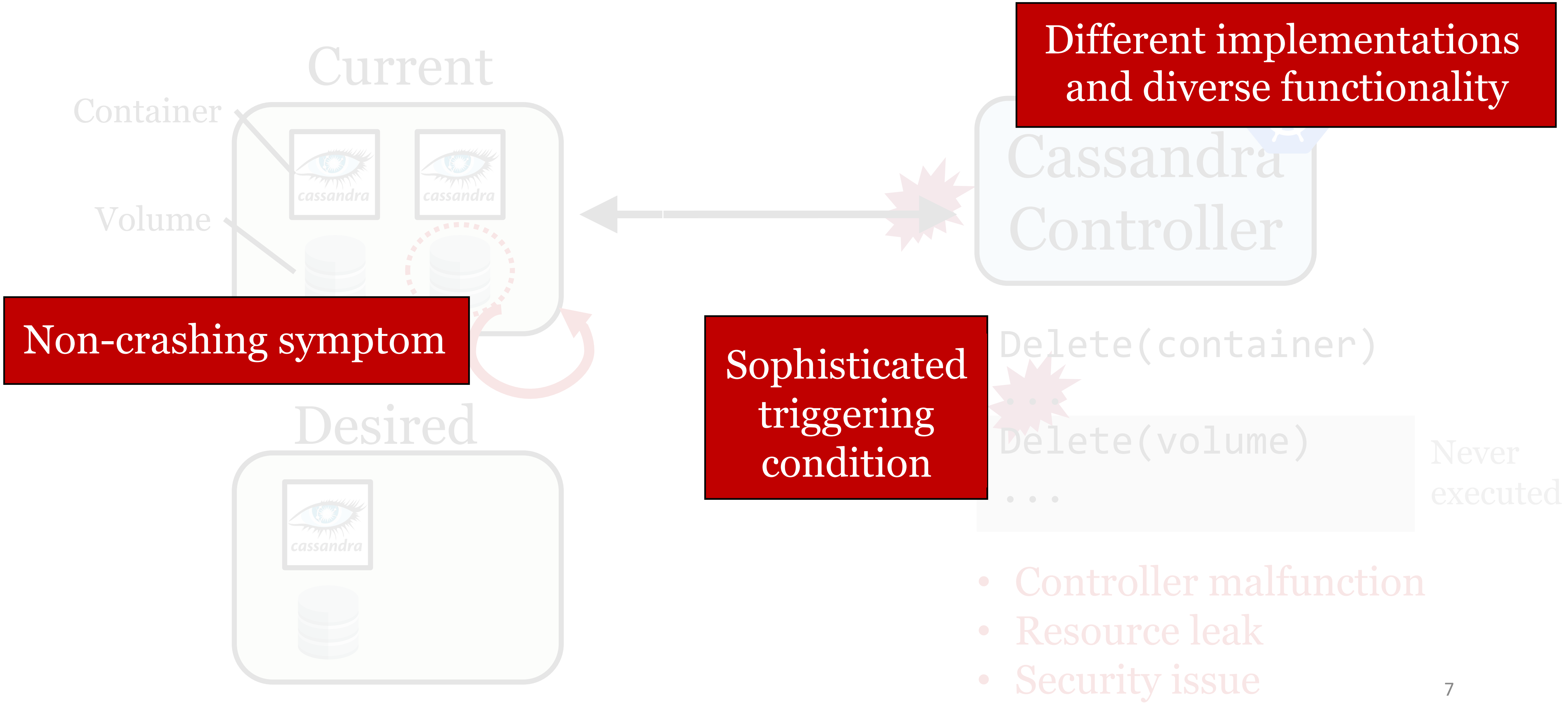
- Sieve: automatic reliability testing for Kubernetes controllers
  - **Key idea:** Perturbing the controller's view of cluster state
  - **Usability:** Testing unmodified controllers
  - **Reproducibility:** Reproducing detected bugs reliably
  - **Open sourced** at https://github.com/sieve-project/sieve

- Detected 46 serious bugs in 10 popular Kubernetes controllers
  - **Severe consequences:** System outage, data loss, security issues, etc.
  - 35 confirmed and 22 fixed

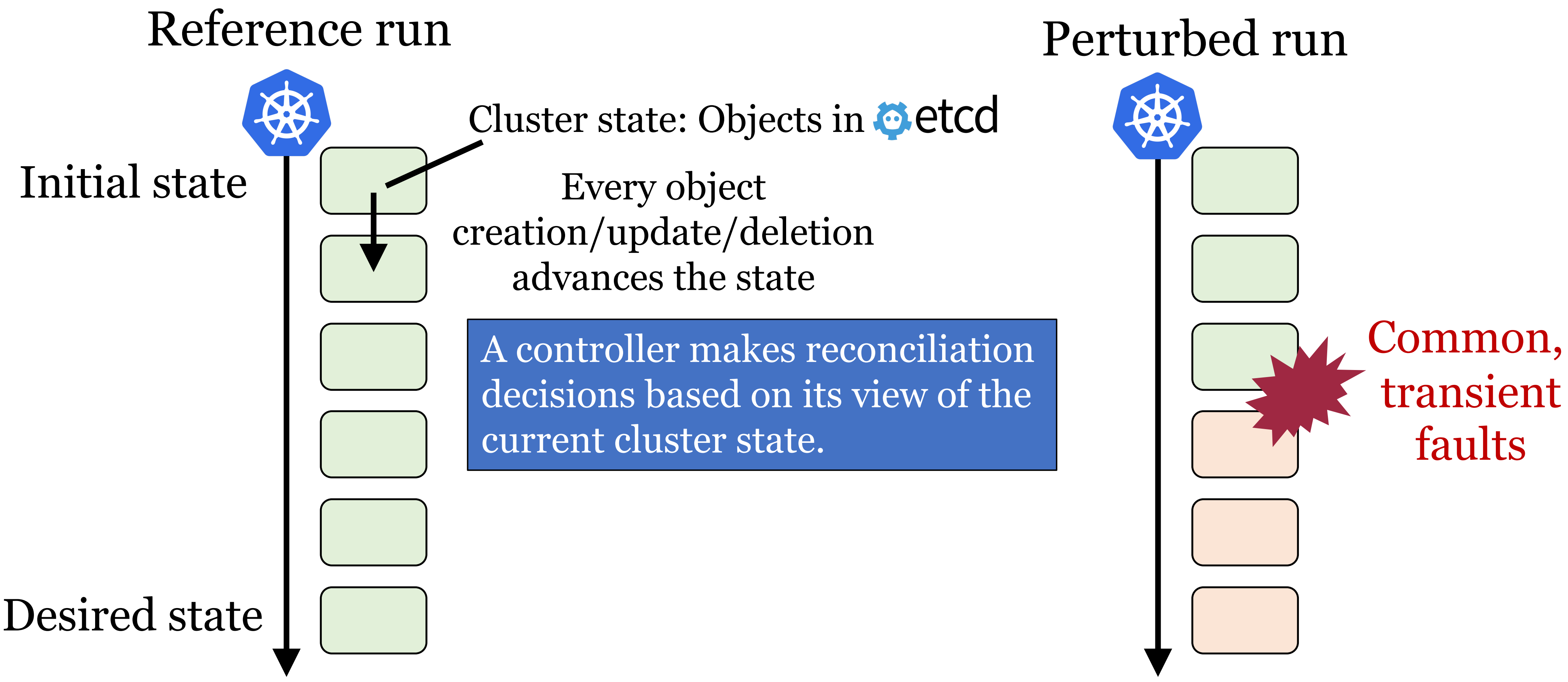
P.H.

GitHub

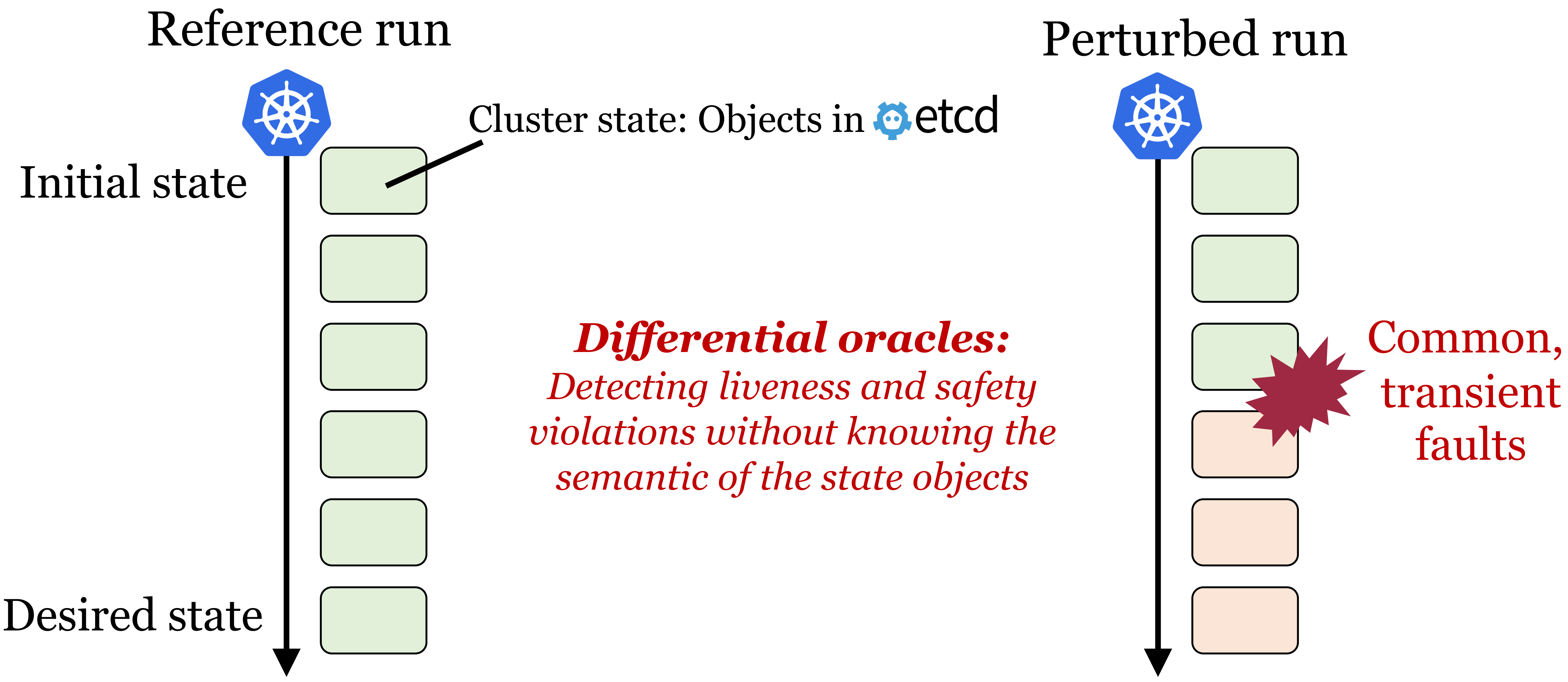
Challenges of testing controllers



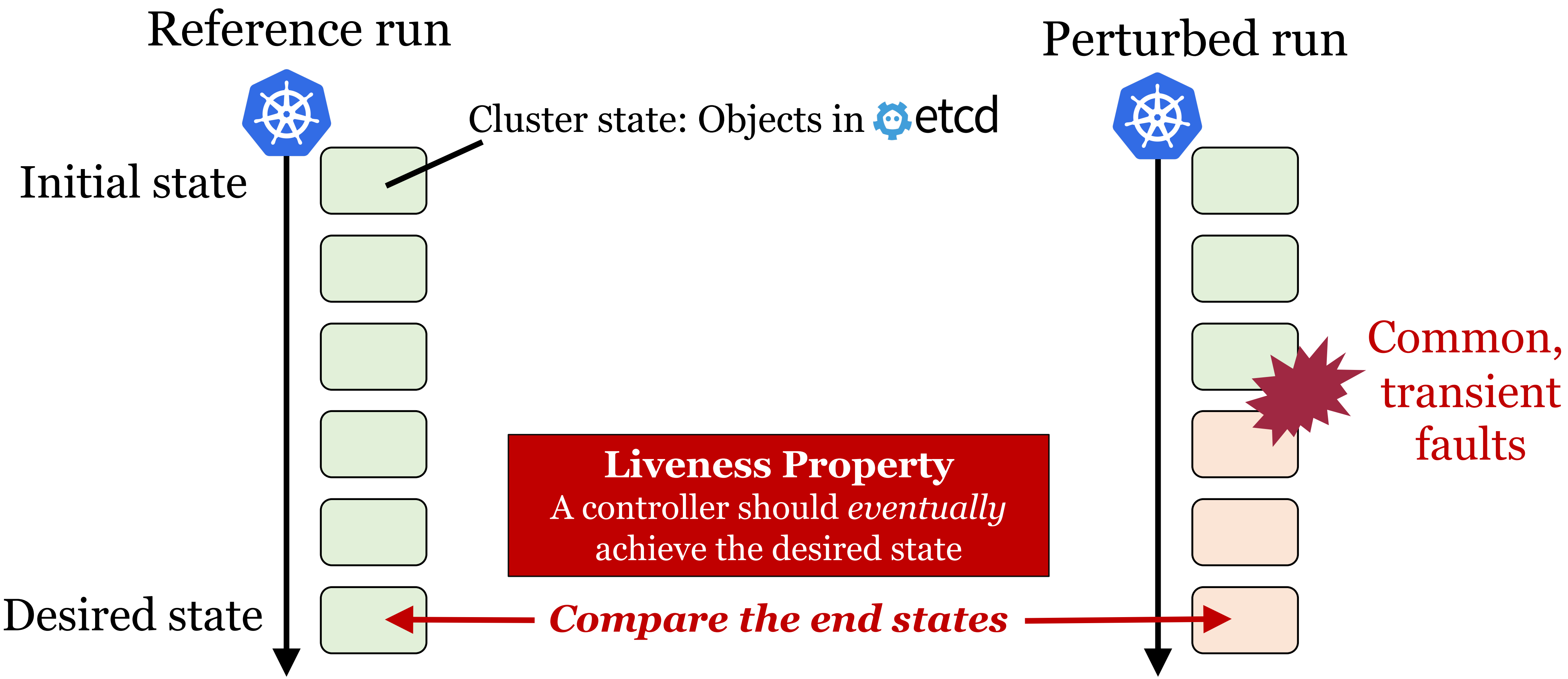
Perturb the controller's view of cluster state



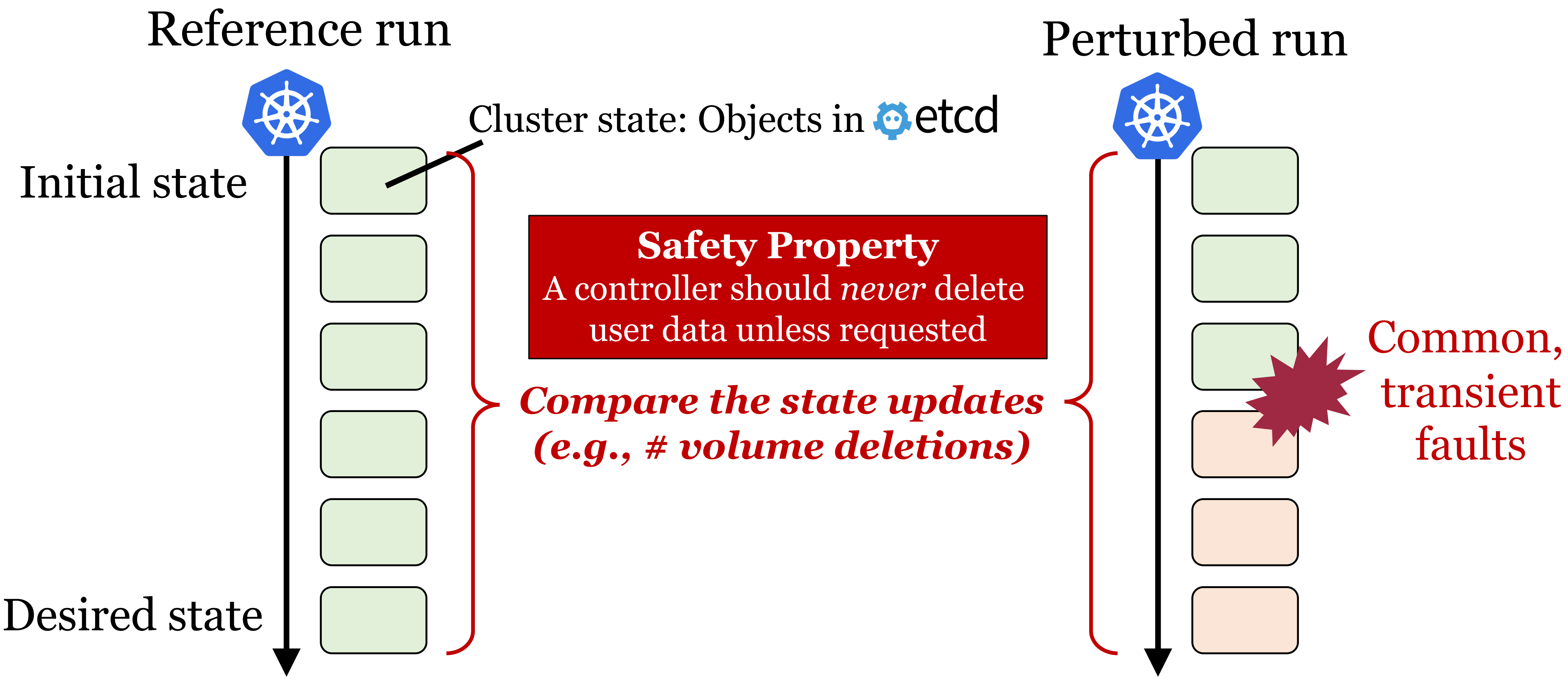
Flag buggy behavior with *differential oracles*



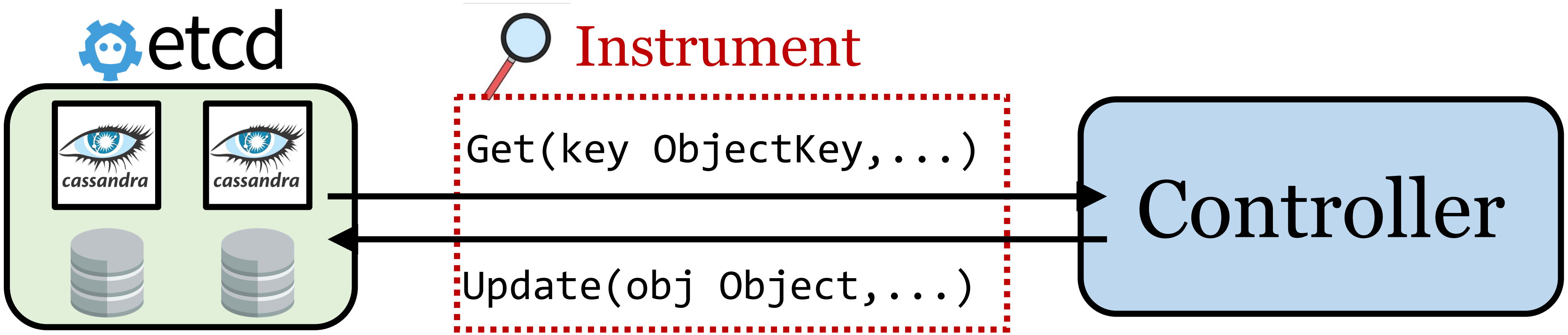
Flag buggy behavior with *differential oracles*



Flag buggy behavior with *differential oracles*



Interposition around state-centric interface

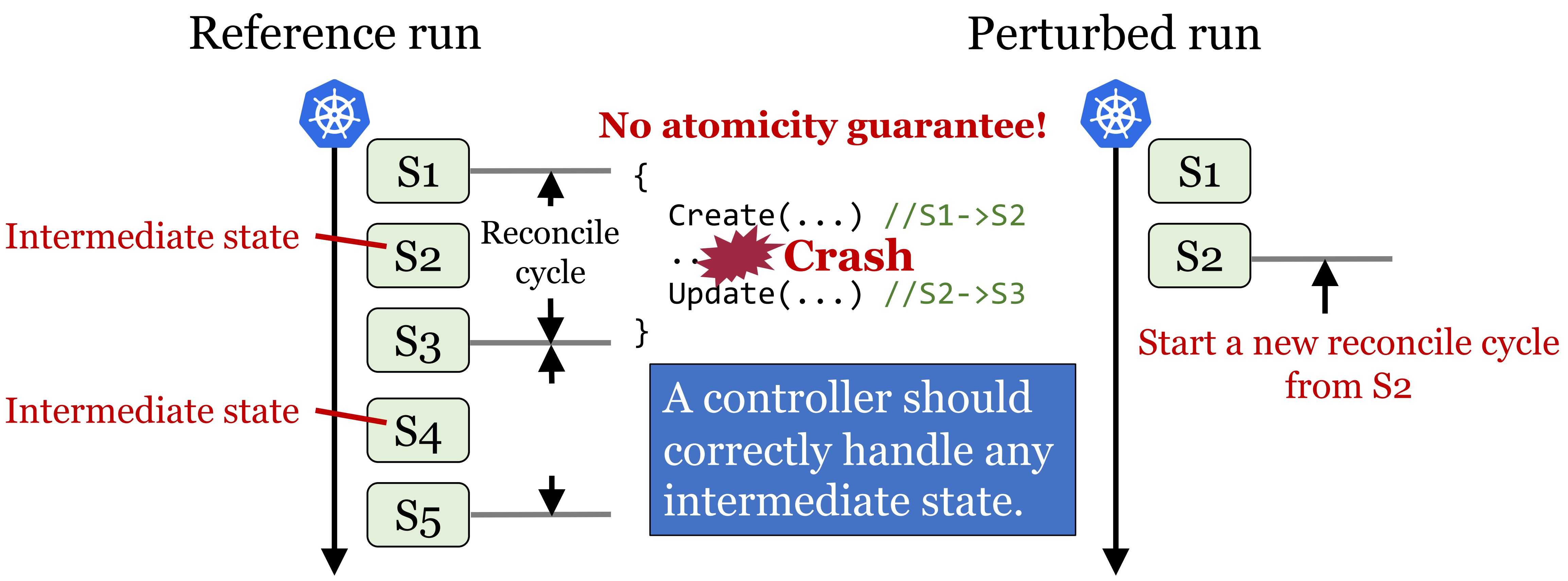


- State-centric interface is used to read/write state objects
- Automatic interposition around cluster state transitions
- Allow Sieve to test ***unmodified*** controllers

Detect diverse controller bugs

- Employ **three perturbation patterns**
- **Exhaustively** test all bug-triggering perturbations
  - Systematically find all the targeted bugs
  - Inject faults with different timings
- Prune out ineffective perturbations to be **efficient**
  - Not every perturbation leads to bugs

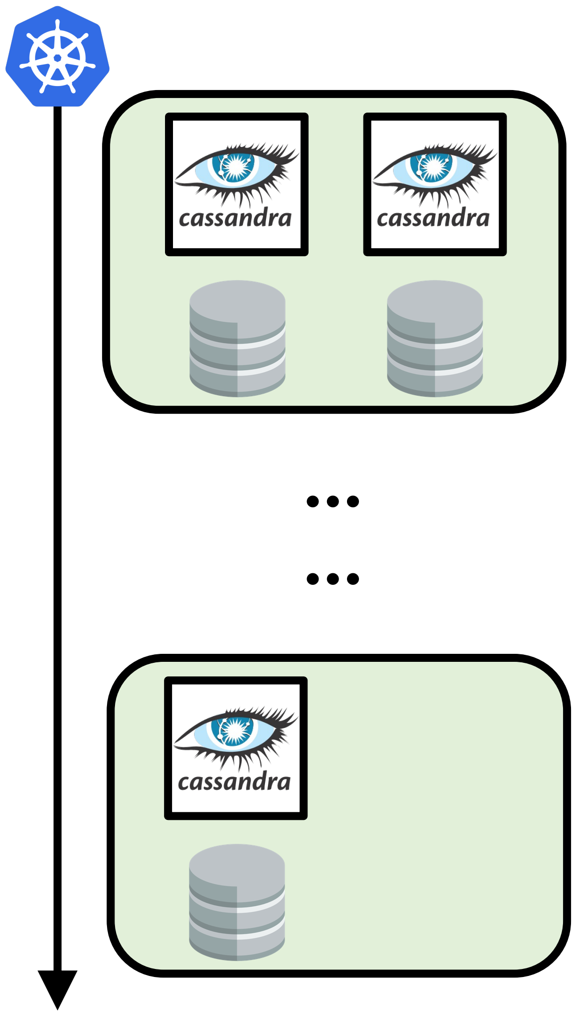
The intermediate-state pattern



A controller should correctly handle any intermediate state.

An intermediate-state bug detected by Sieve

Reference run

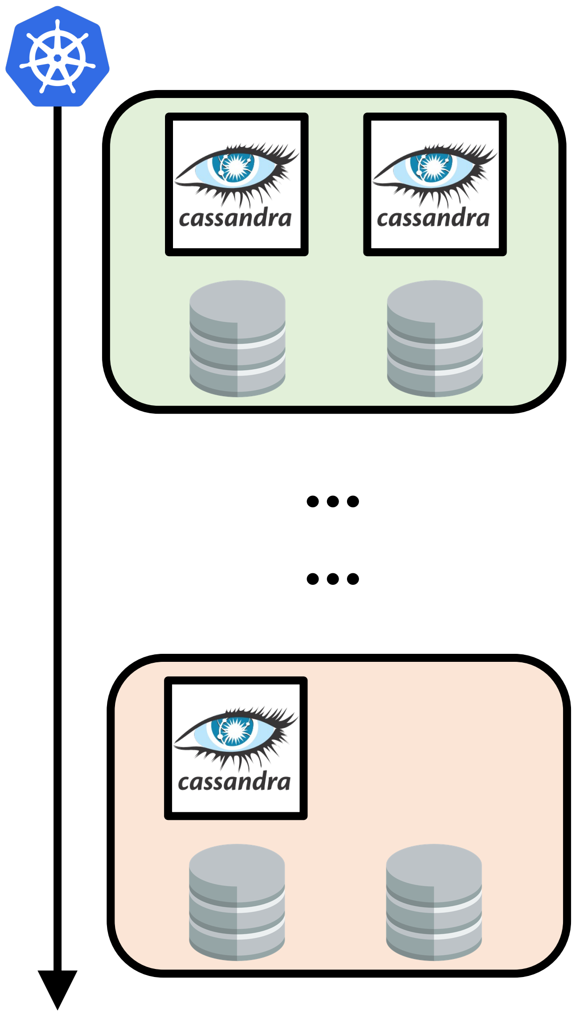


Uses phase to drive the reconciliation

```
switch phase {
case “Ongoing”:
    if ContainerNotFound(container) {
        return Error(“Container not found”)
    }
    ...
    Delete(container)
    ...
    UpdatePhase(“Finalizing”) // phase <- “Finalizing”
    ...
case “Finalizing”:
    ...
    Delete(volume)
    ...
    UpdatePhase(“Done”) // phase <- “Done”
}
// https://github.com/Orange-OpenSource/casskop
```

An intermediate-state bug detected by Sieve

Perturbed run



Always returns here…

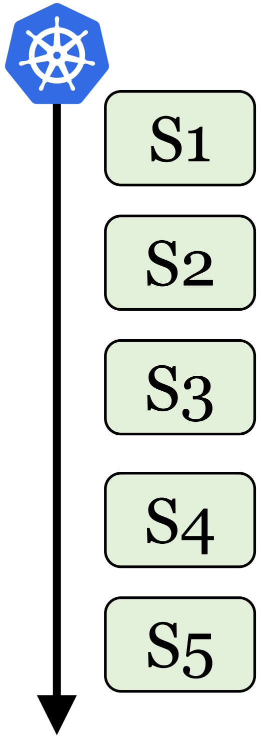
```
switch phase {
case “Ongoing”:
    if ContainerNotFound(container) {
        return Error(“Container not found”)
    }
    ...
    Delete(container)
    ...
    UpdatePhase(“Finalizing”) // phase <- “Finalizing”
    ...
case “Finalizing”:
    ...
    Delete(volume)
    ...
    UpdatePhase(“Done”) // phase <- “Done”
}
// https://github.com/Orange-OpenSource/casskop
```

Crash

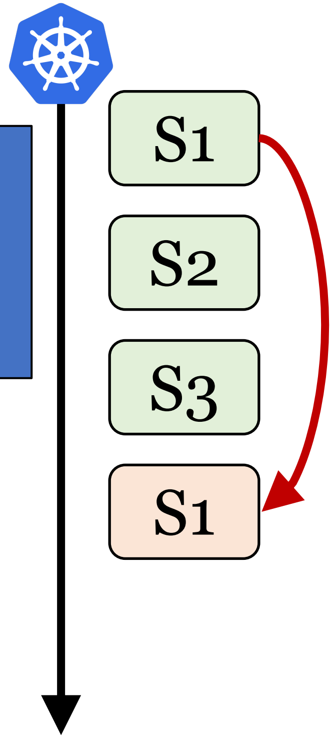
Never executes this…

The stale-state pattern

Reference run



Perturbed run

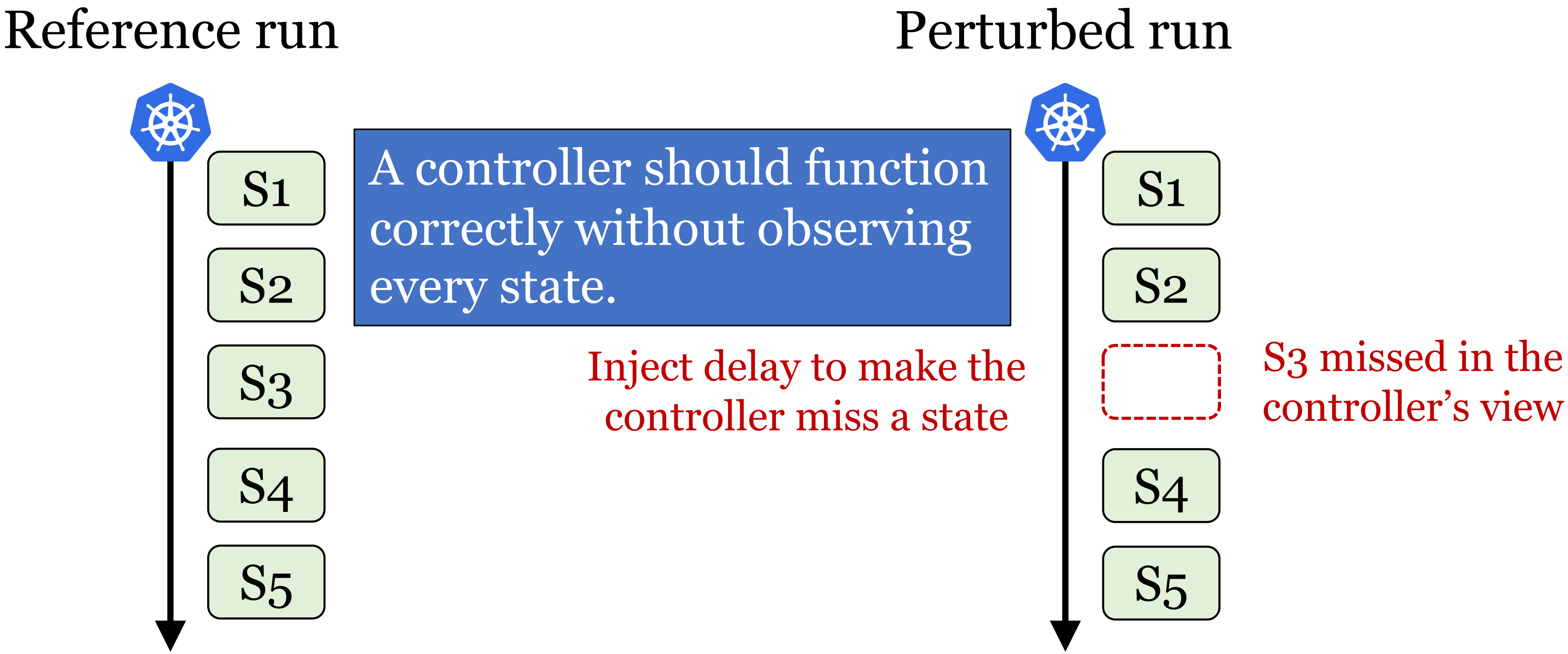


S1 replayed in the controller's view

A controller should correctly handle staleness caused by asynchrony and caching.

1. Inject delay to make a backup cache stale
2. Reconnect the controller to the stale cache

The unobserved-state pattern



Exhaustive perturbation for each pattern

- **Key principle:** Inject faults at each execution point

- Run many different tests, each performing a different perturbation
  - Intermediate-state: Crash after ***every*** state update
  - Stale-state: Replay ***every*** stale state
  - Unobserved-state: Make the controller miss ***every*** state

Prune ineffective perturbations for efficiency

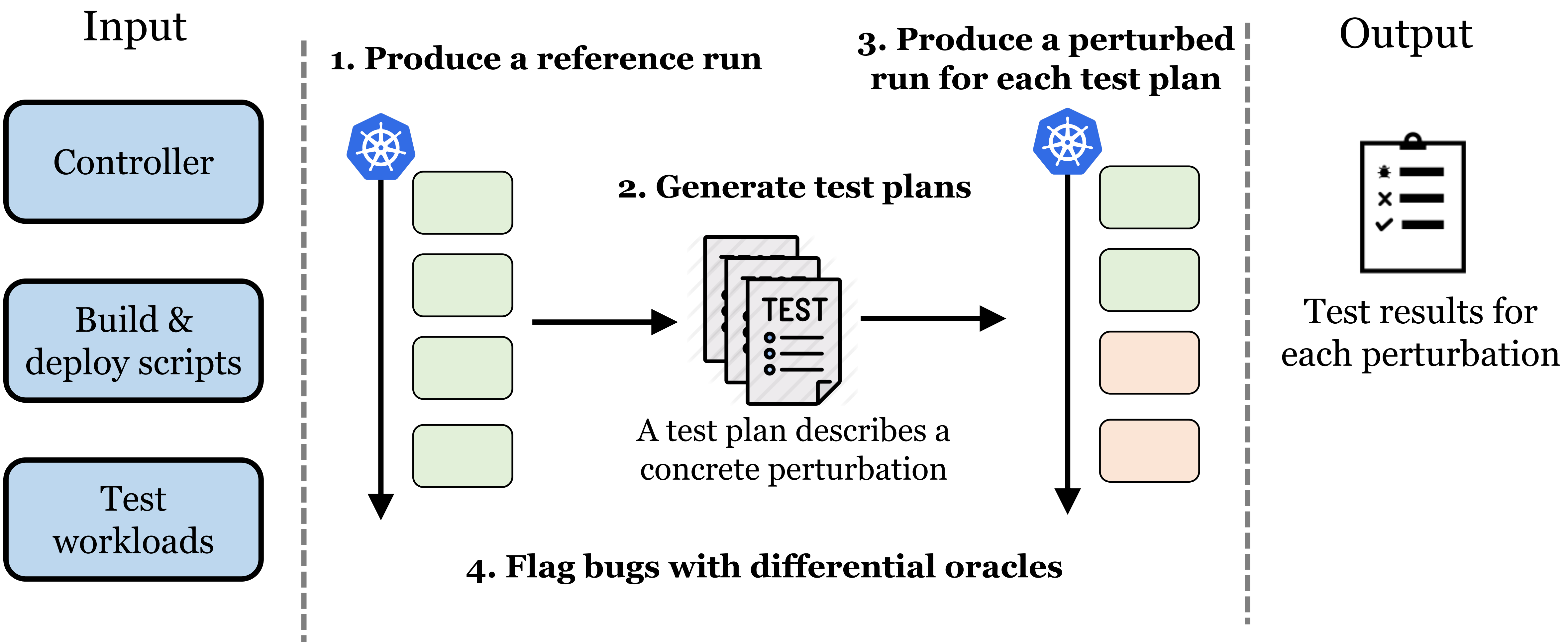
Key principle: Prune out perturbations that cannot affect a controller's behavior

- **Intermediate-state:** Prune out crashes that do not result in new intermediate states
- **Stale- and unobserved-state:** Avoid perturbing the state if observing the state does not causally lead to any controller effect
  - Reason about **causality** from state to effect

```
// Reconcile cycle
...
Create(...)
...
Update(...)
...
Delete(...)
...
Update(...)
...
```

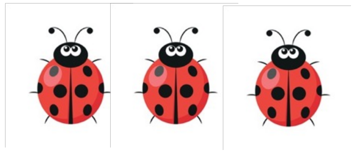
Delete a non-existing object

Sieve end-to-end workflow



Evaluation

- Applied Sieve to **10** popular Kubernetes controllers
- Can Sieve **effectively** find new bugs in real-world controllers?
  - Sieve found **46** bugs in **10** controllers
- Does Sieve do so **efficiently**?
  - Sieve pruned out **46% - 99%** of perturbations
  - Sieve tested each controller with a **nightly** run
- Are Sieve's testing results **trustworthy**?
  - Sieve had a low false positive rate of **3.5%**

Finding *new* bugs 

35 confirmed; 22 fixed

Controller	Intermediate state bugs	Stale state bugs	Unobserved state bugs	Indirect bugs	Total
cass-operator	2	1	0	0	3
cassandra-operator	0	2	1	2	5
casskop	1	2	1	0	4
elastic-operator	0	2	0	0	2
mongodb-operator	2	3	1	3	9
nifikop	2	0	0	1	3
rabbitmq-operator	1	2	1	0	4
xtradb-operator	3	3	1	0	7
yugabyte-operator	0	2	1	2	5
zookeeper-operator	0	2	1	1	4
Total	**11**	**19**	**7**	**9**	**46**

Conclusion

- Controller reliability is critical but challenging!

- Sieve: automatic reliability testing for Kubernetes controllers
  - **Key idea:** Perturbing the controller's view of the cluster state
  - **Usability:** Testing unmodified controllers
  - **Reproducibility:** Reproducing detected bugs reliably

- **Open sourced** at https://github.com/sieve-project/sieve
  - Test your controller with Sieve!