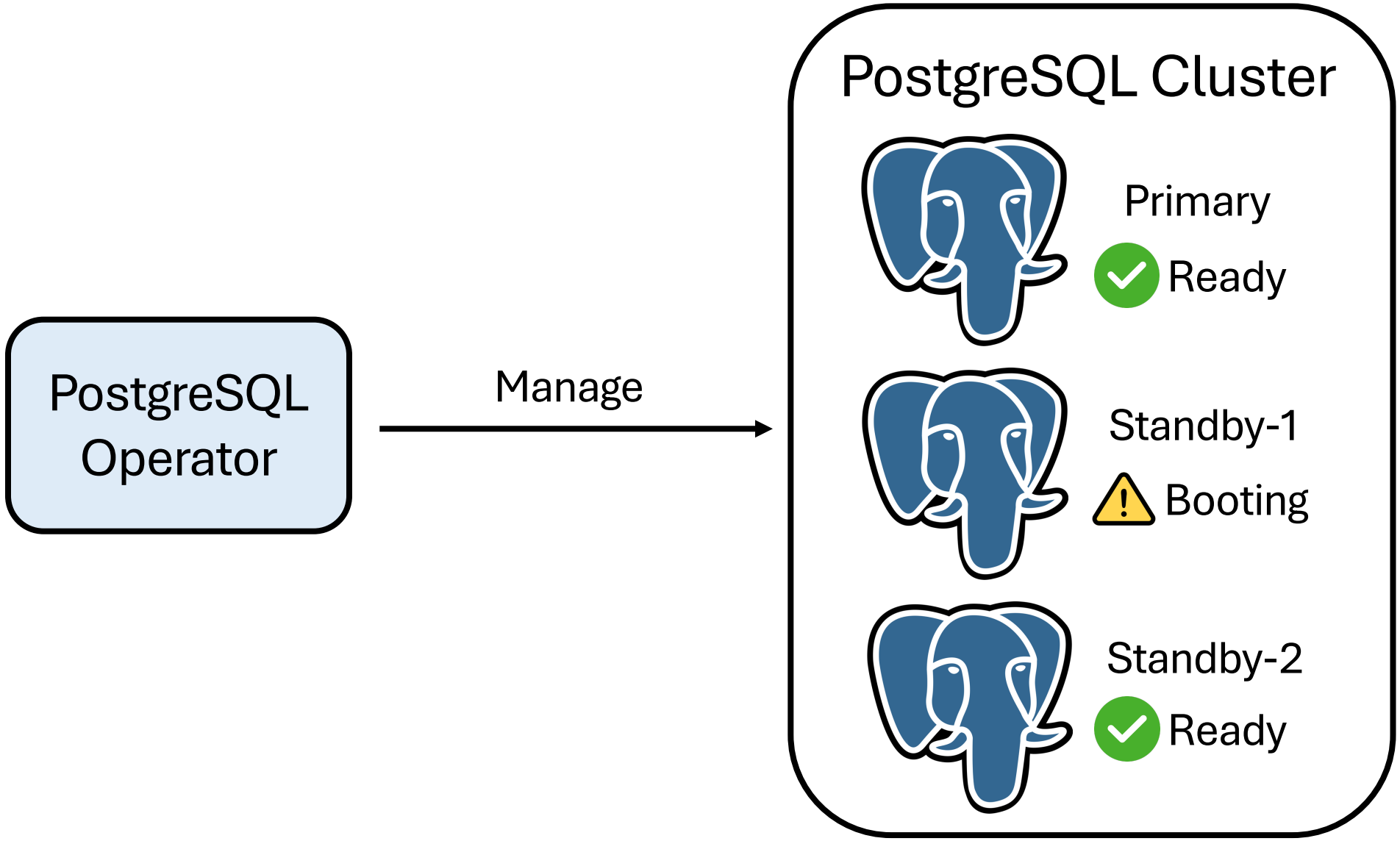
Who Watches the Watchers? On the Reliability of Softwarizing Cloud Application Management

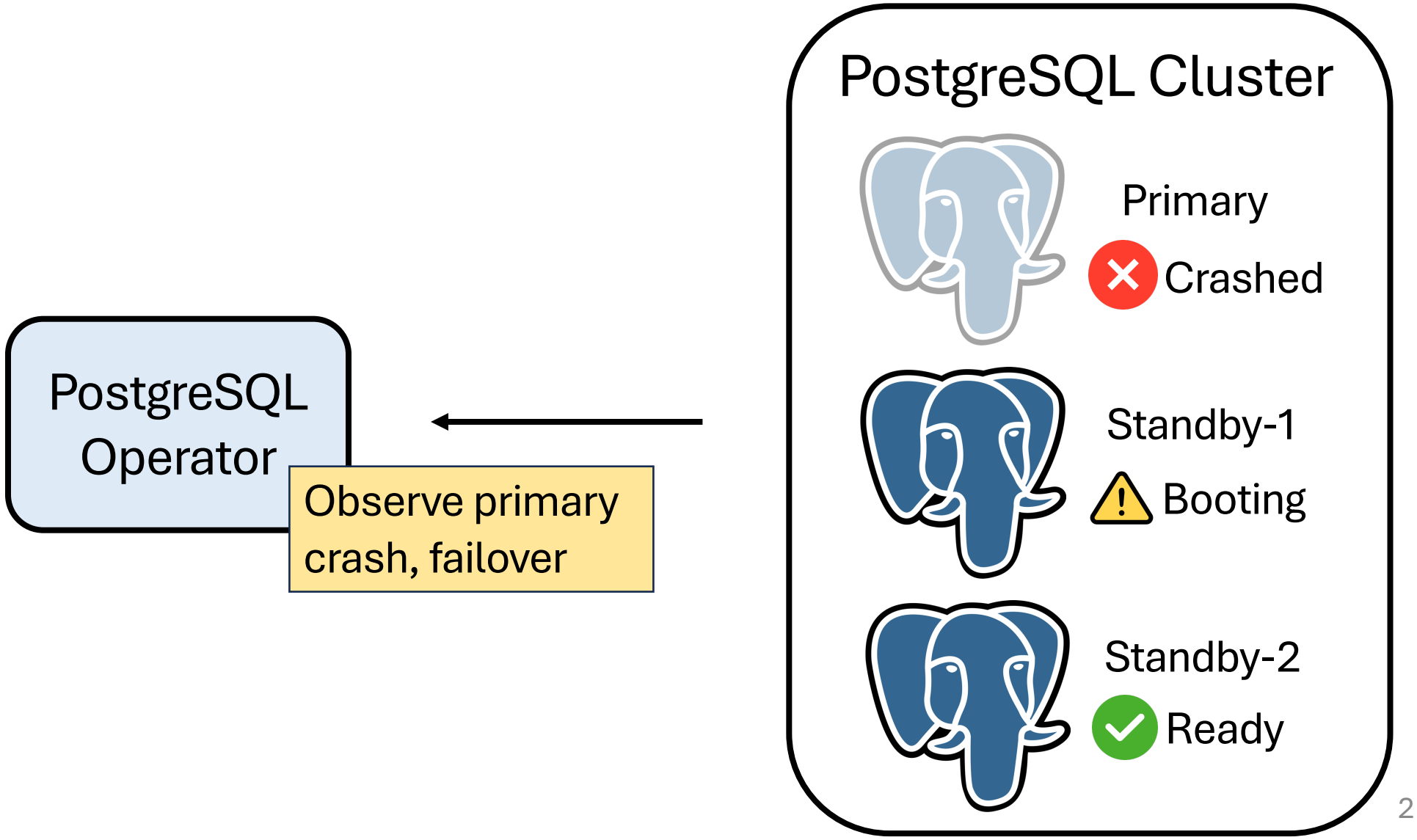
Jiawei Tyler Gu, Zhen Tang, Yiming Su, Bogdan Alexandru Stoica, Xudong Sun
William X. Zheng, Yue Zhang, Akond Rahman, Chen Wang, Tianyin Xu

IBM **Research**

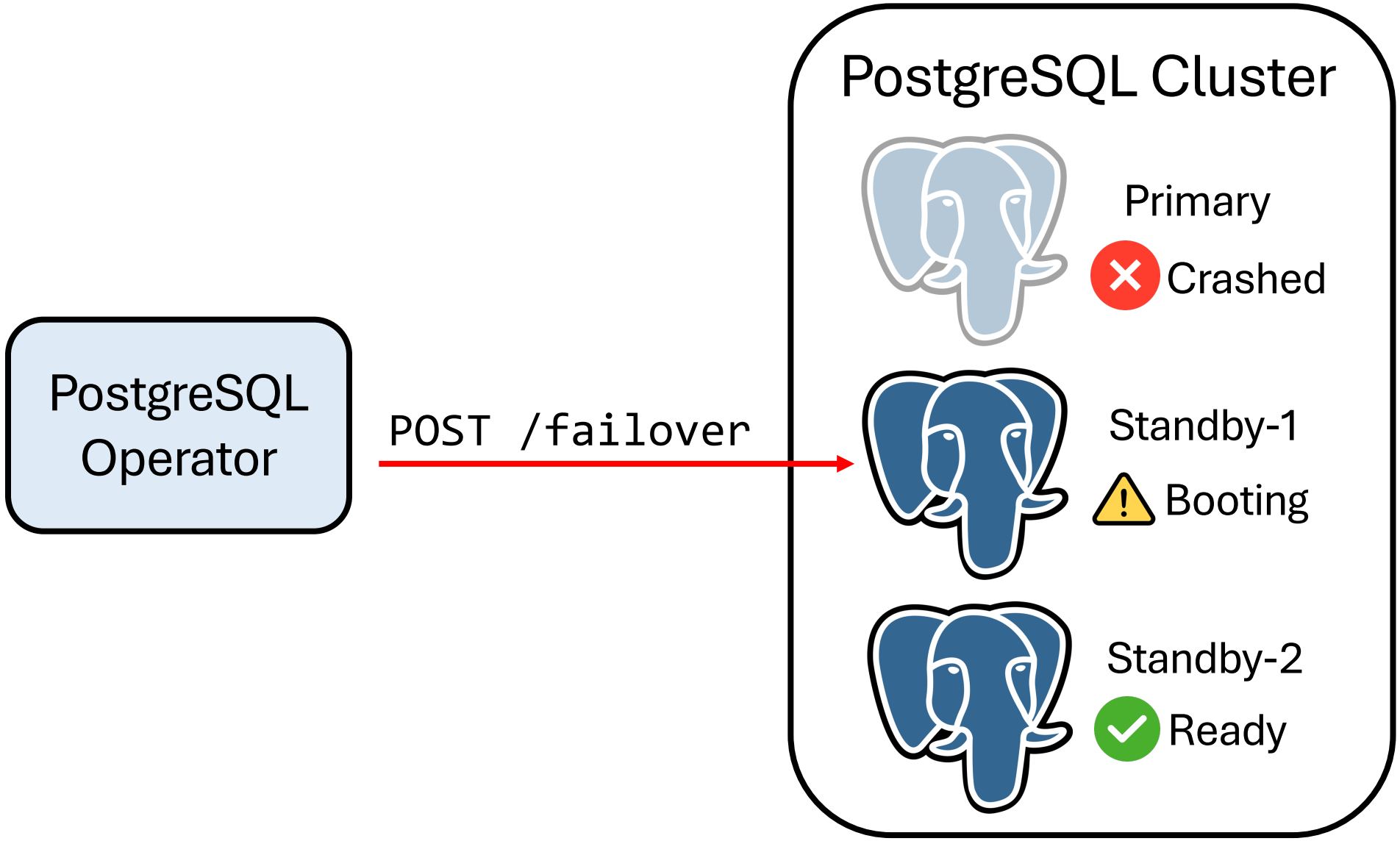
Faulty operations cause production failures



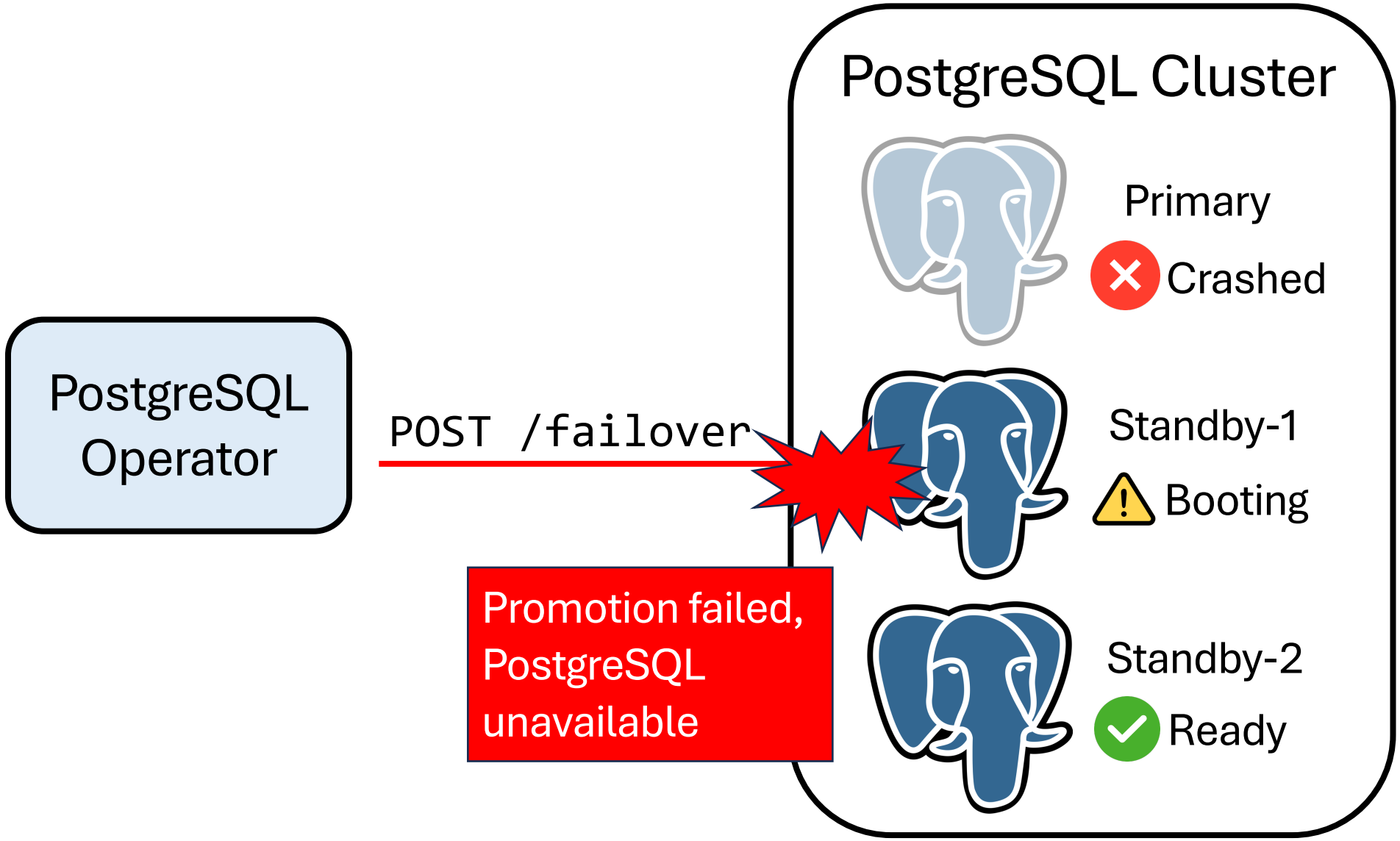
Faulty operations cause production failures



Faulty operations cause production failures



Faulty operations cause production failures



Operation failures are catastrophic

Spaces / Percona Operator for... / K8SPSMDB-956

Certificate Rotation brought the Sharded MongoDB cluster down

Significant performance degradation in Kafka JMX metrics on upgrade #7943

Closed **neilclark-bgn** opened this issue on Jan 19 · 1 comment

Bootstrapped security.json incorrectly assigns the "all" permission to the "users" role due to ordering of rules #274

Closed #299

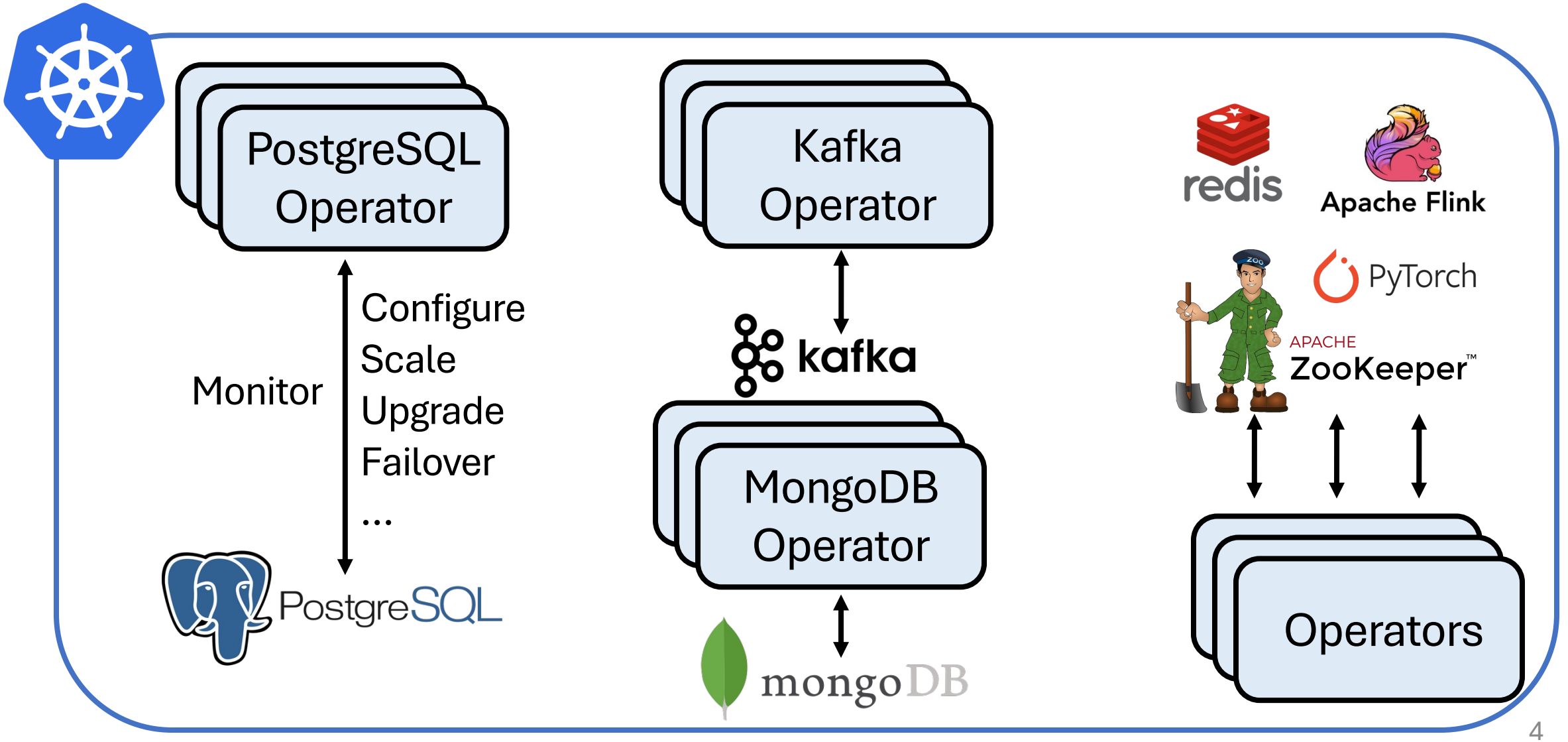
Applications are managed by softwarized operators

PostgreSQL Operator

Monitor

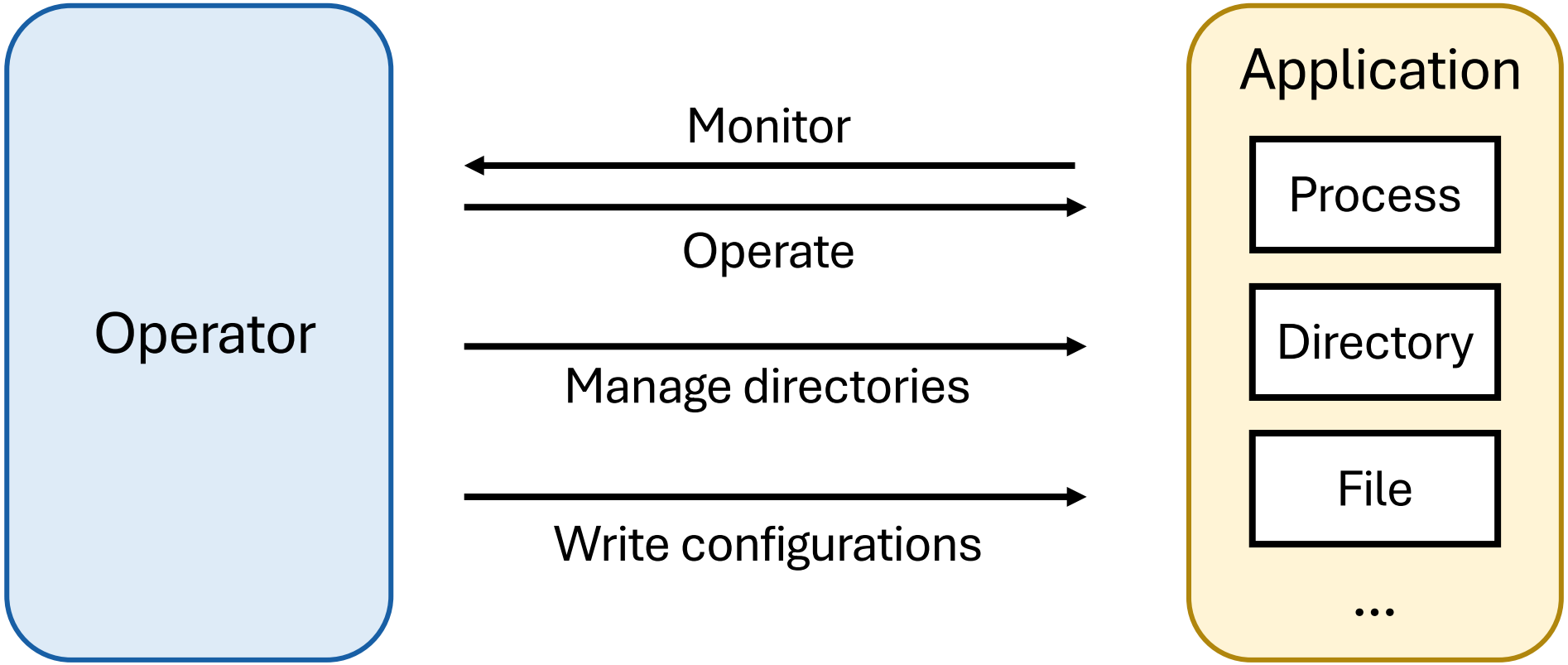
Configure
Scale
Upgrade
Failover
...

Applications are managed by softwarized operators



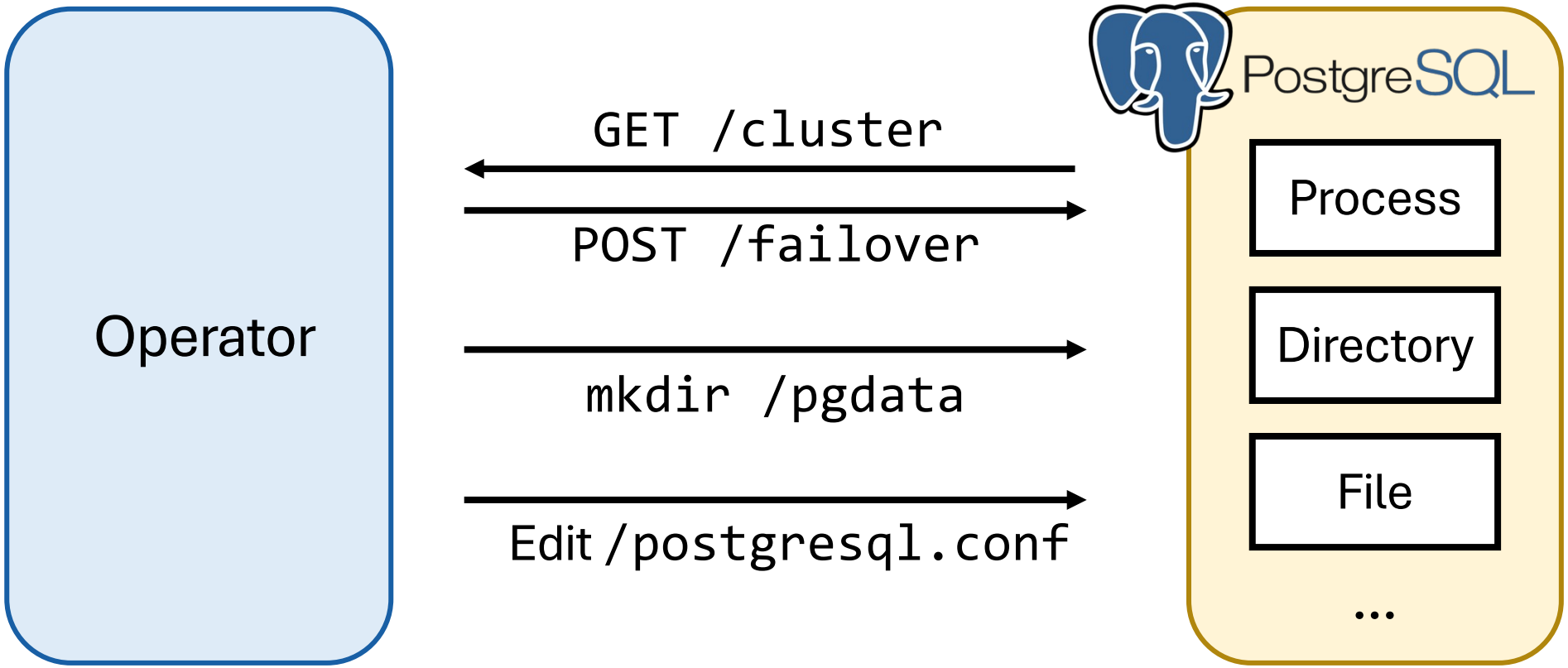
Operators interact with four external entities

1. The managed application



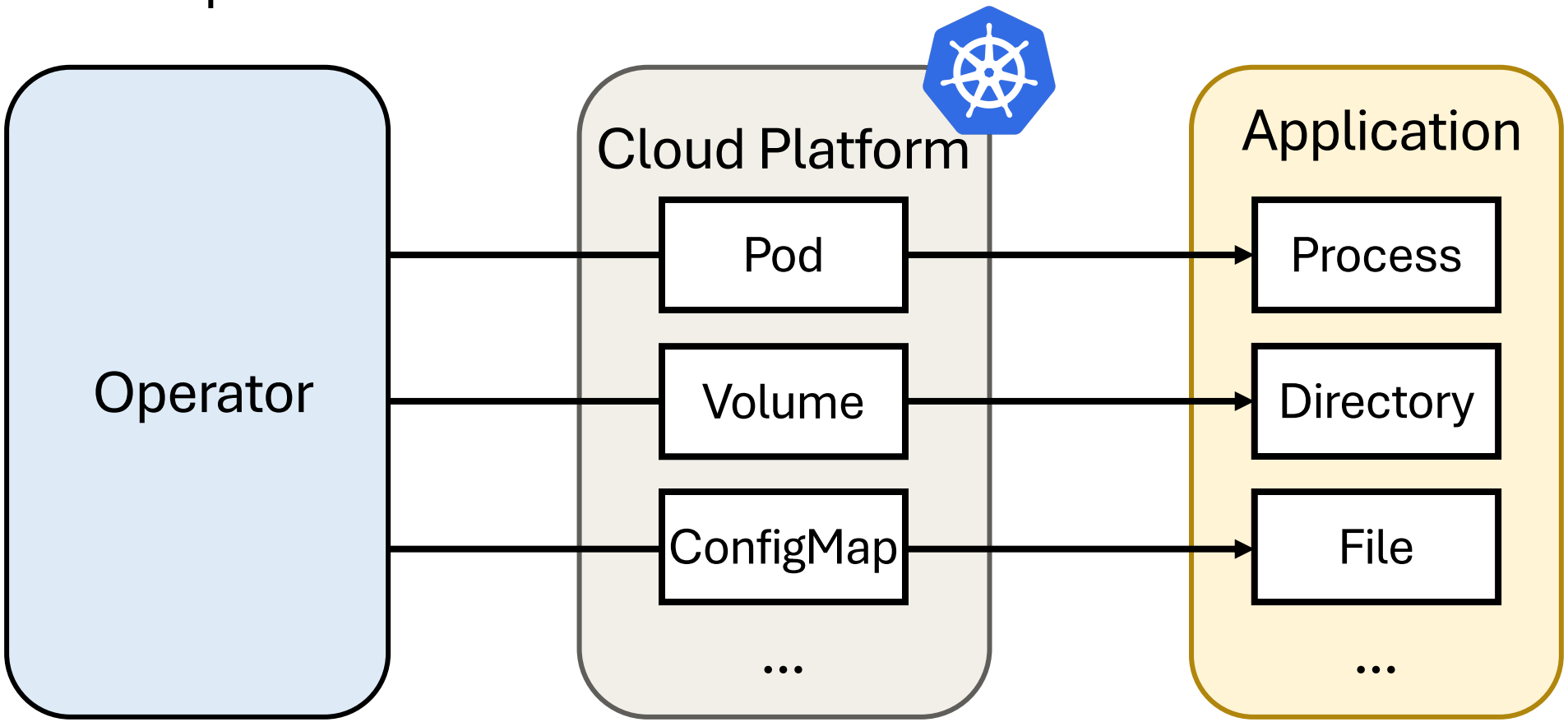
Operators interact with four external entities

1. The managed application



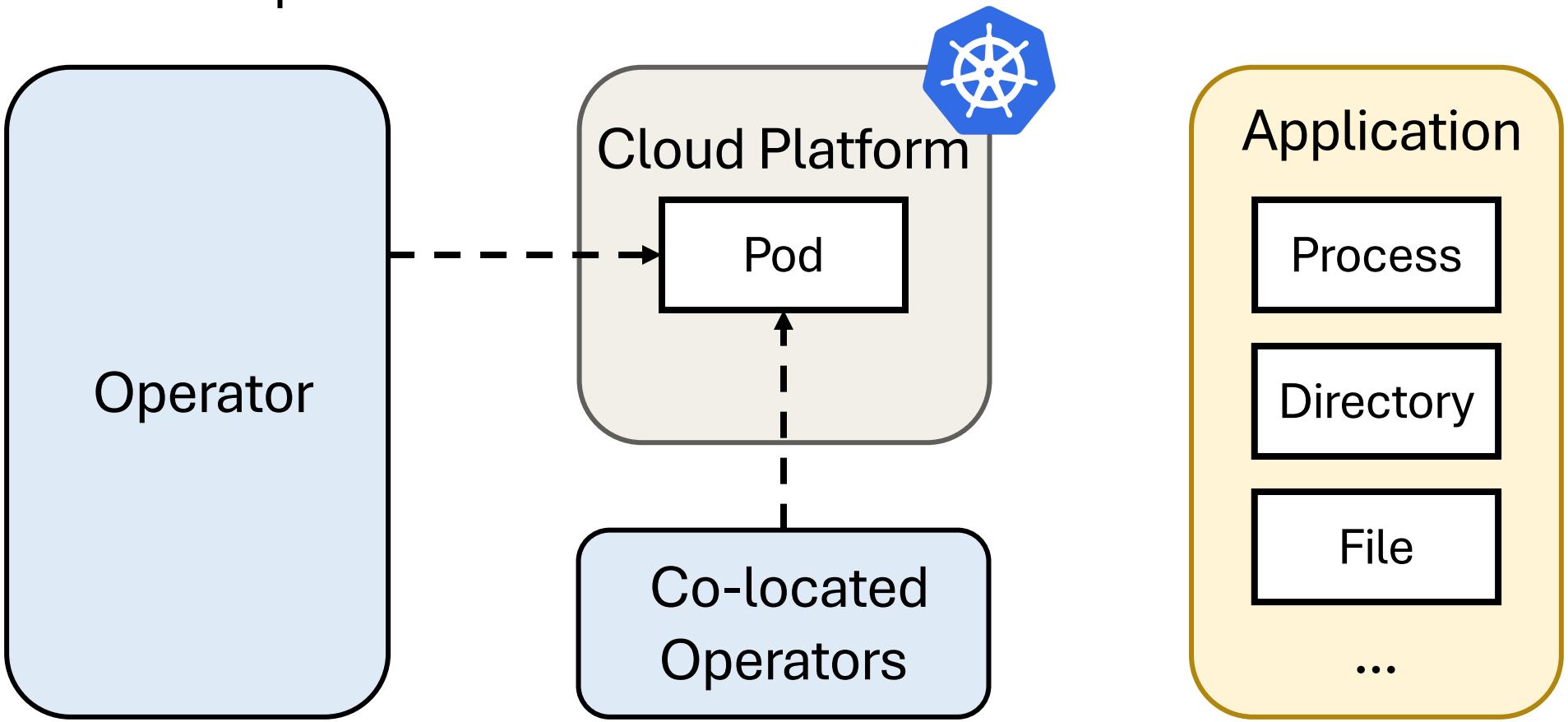
Operators interact with four external entities

2. The cloud platform



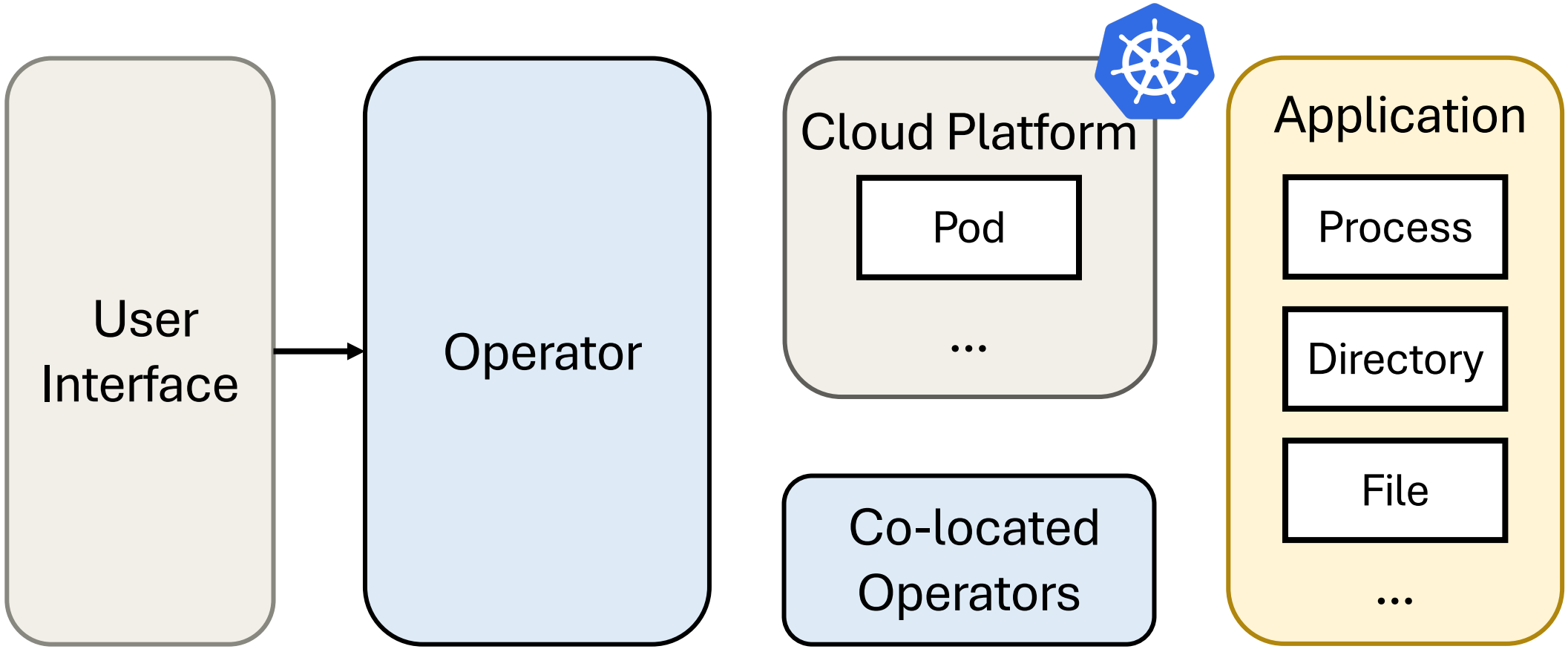
Operators interact with four external entities

3. Co-located operators

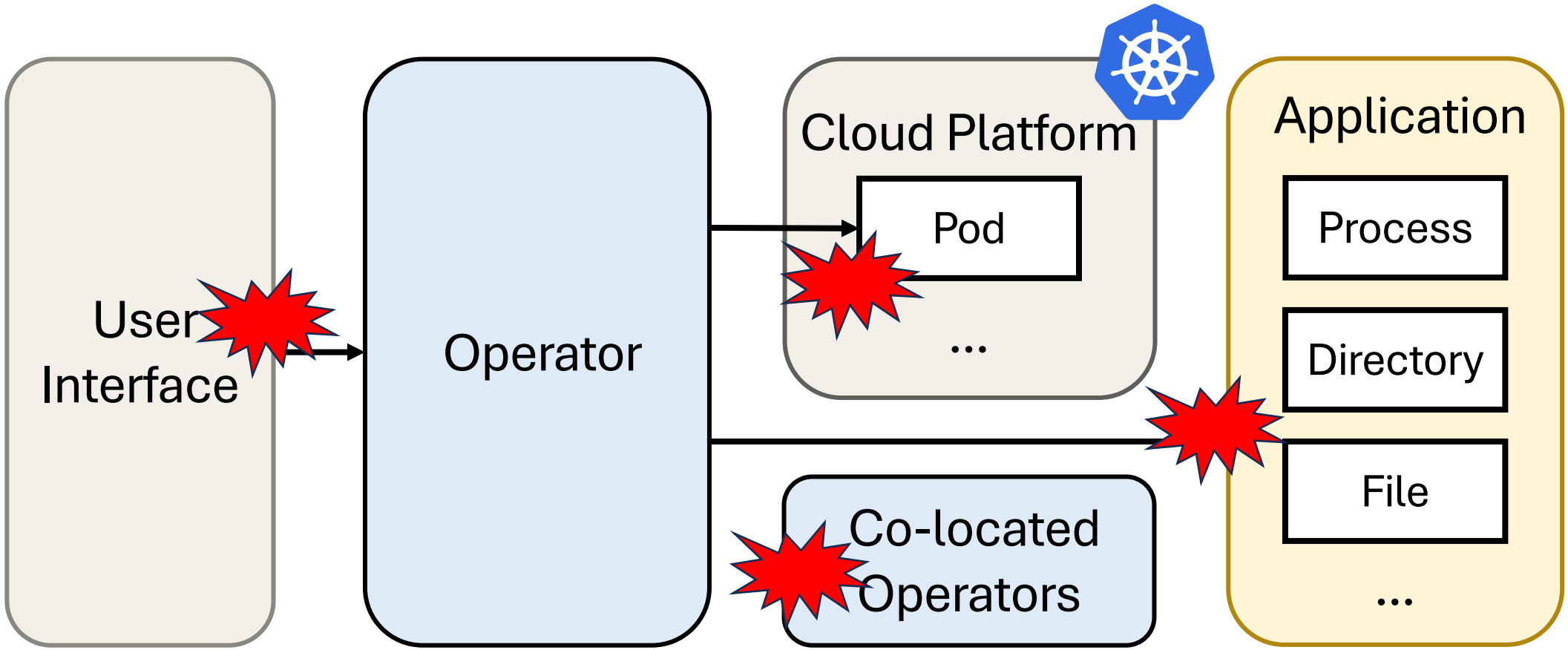


Operators interact with four external entities

4. User Interface



Operators interact with four external entities



Contributions

- A study of 412 real-world operator failures
  - Operator-application interactions are the largest yet overlooked
- Oat: an automatic testing tool for operator-application interactions
  - Push-button for unmodified Kubernetes operators
  - Now part of Acto: https://github.com/xlab-uiuc/acto
- Detected **86 new bugs** in 6 popular Kubernetes operators
  - 53 confirmed, 28 fixed

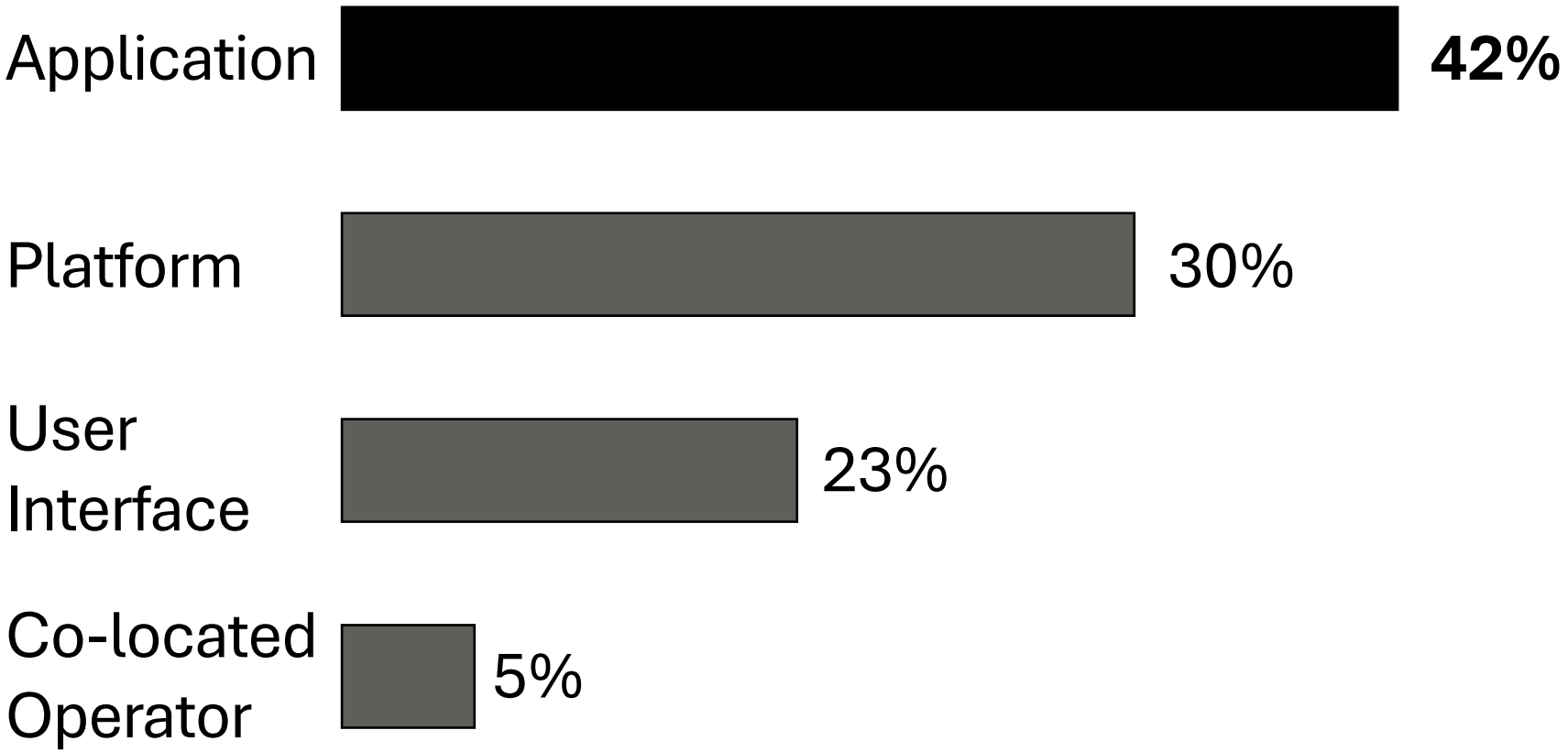
An empirical study on operator failures

- 412 real-world failures sampled from 13 mature Kubernetes operators
  - Managing *server applications, distributed systems, platform runtimes*

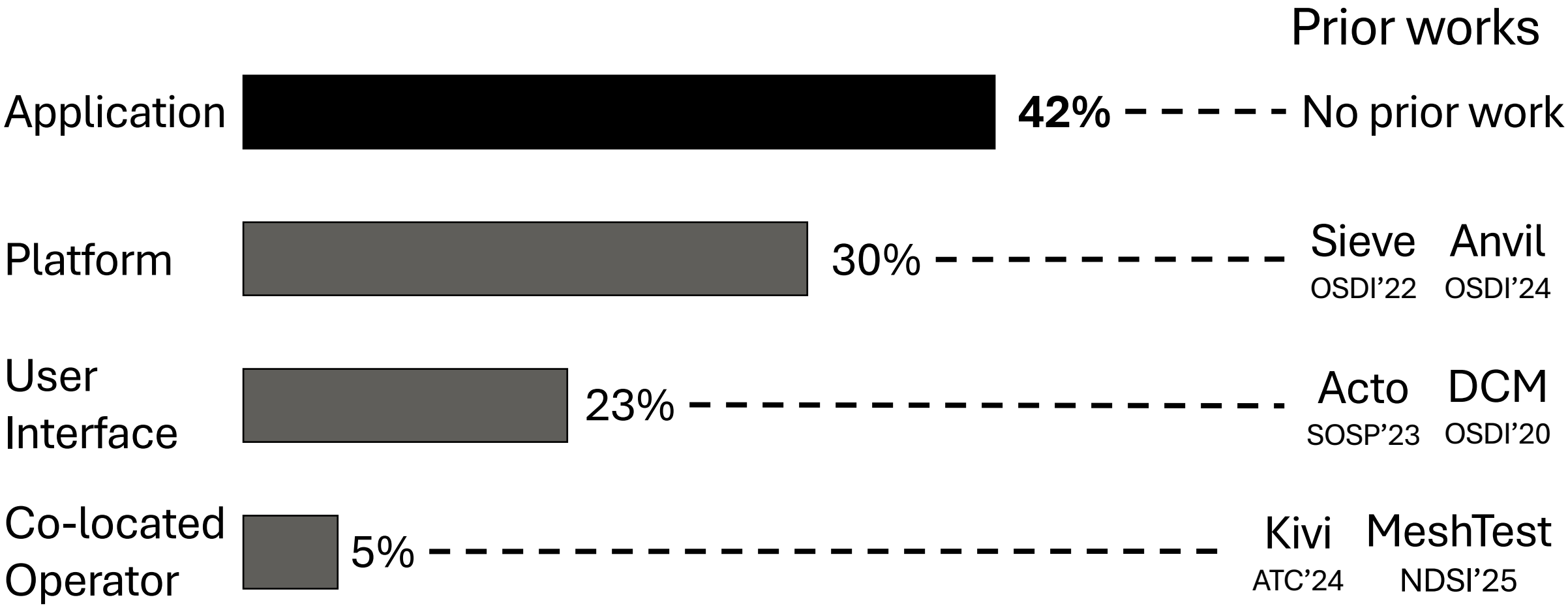
- Analyze root causes and consequences
  - Each case reviewed by two authors

Interaction failures dominate (52%)

Application interaction is the largest, yet overlooked



Application interaction is the largest, yet overlooked



Four patterns of operator-application failures

Semantic violations

Operator breaks application's operation semantics

64%

Four patterns of operator-application failures

Semantic violations

Operator breaks application's operation semantics

64%

State Observability 16%

Fail to observe application internal state

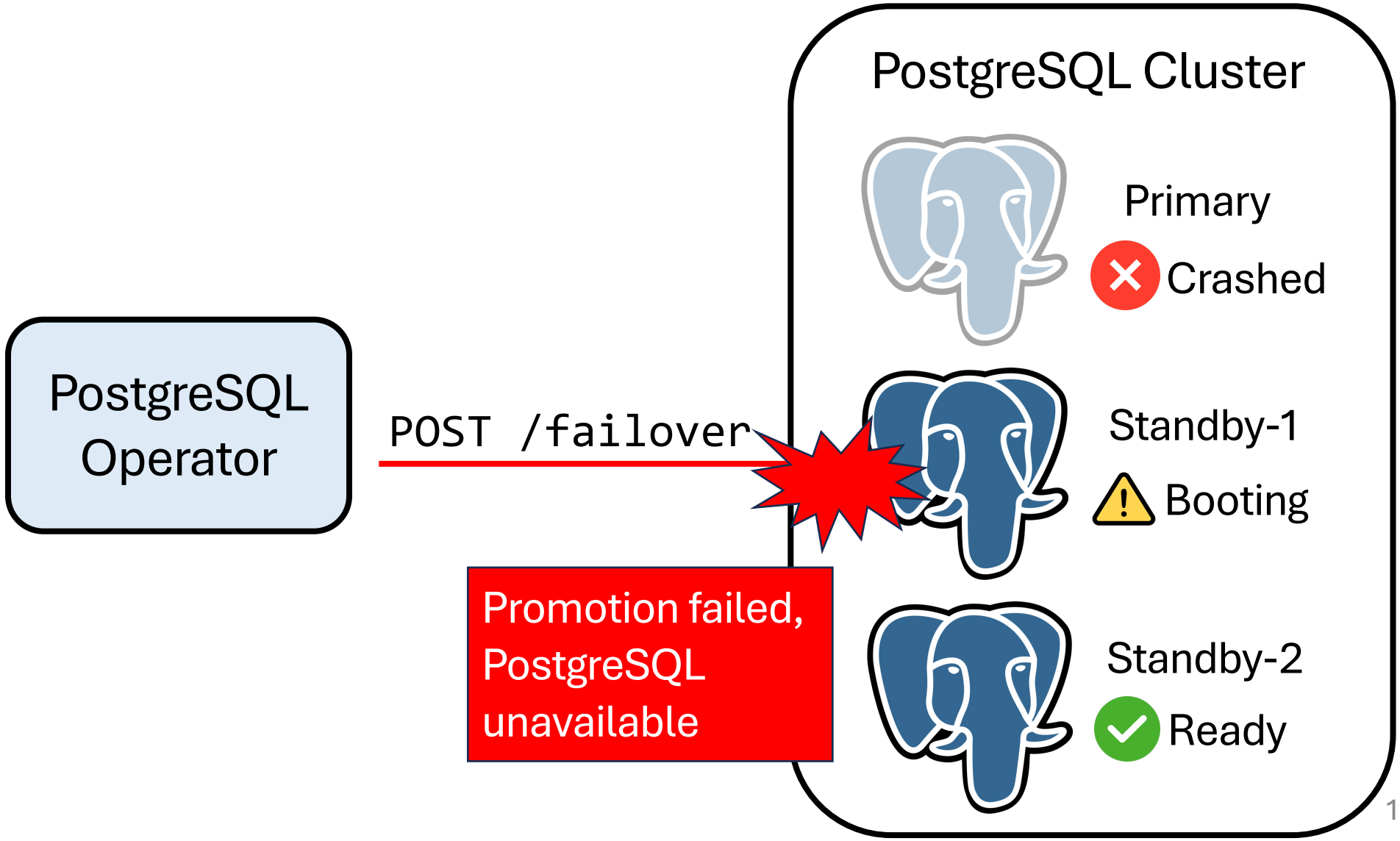
Version Incompatibility 12%

Fail to handle inconsistent behavior between versions

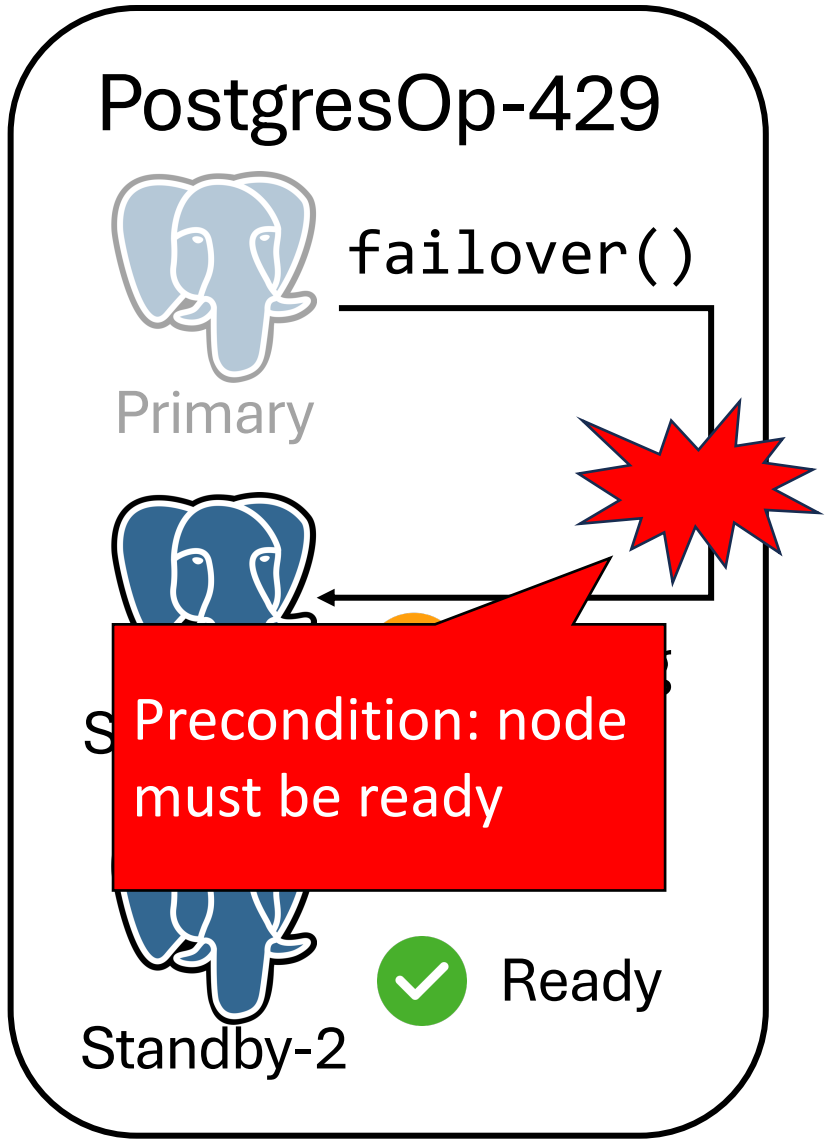
Error Handling 8%

Fail to handle application errors

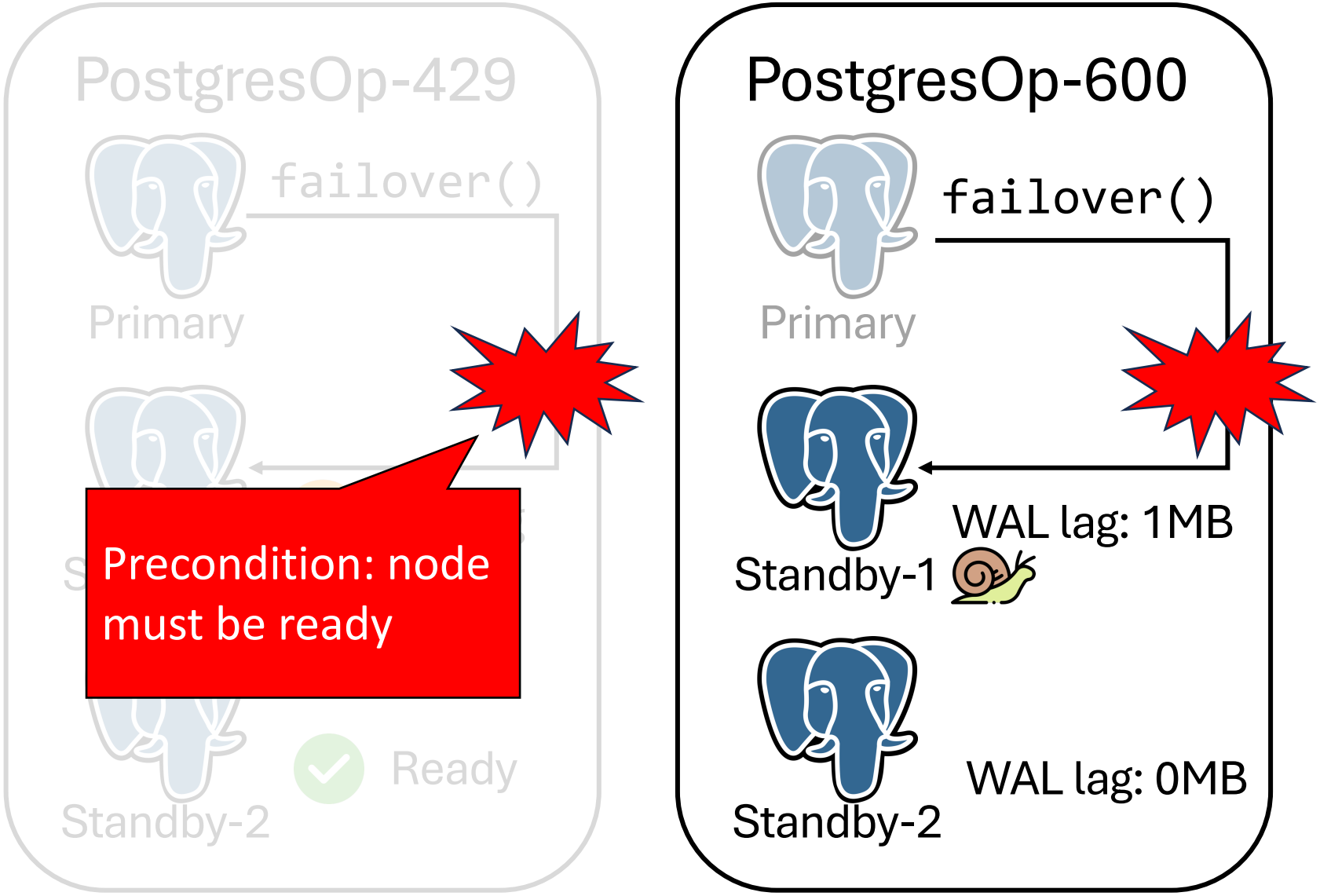
Preconditions are often unknown until violated



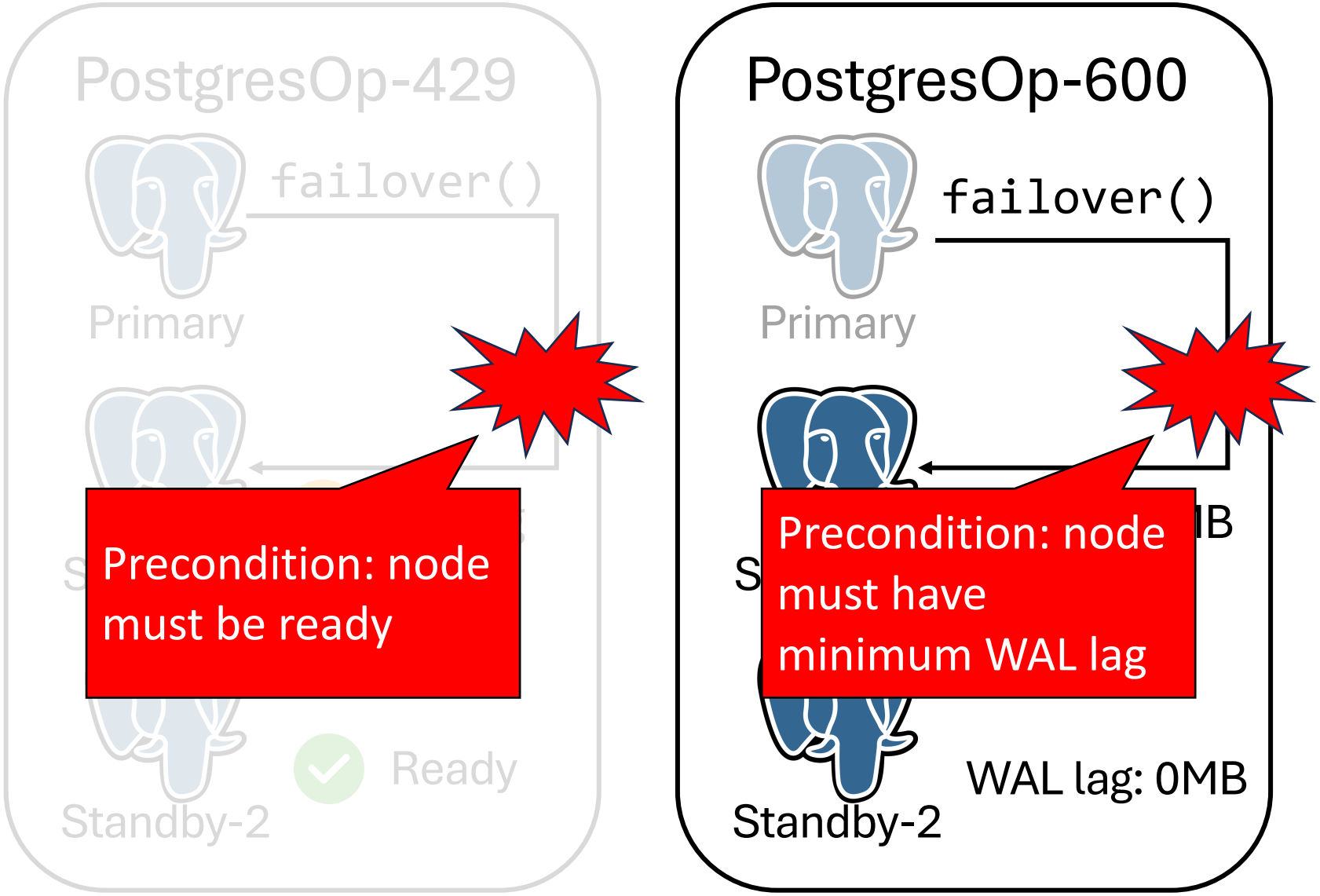
Preconditions are often unknown until violated



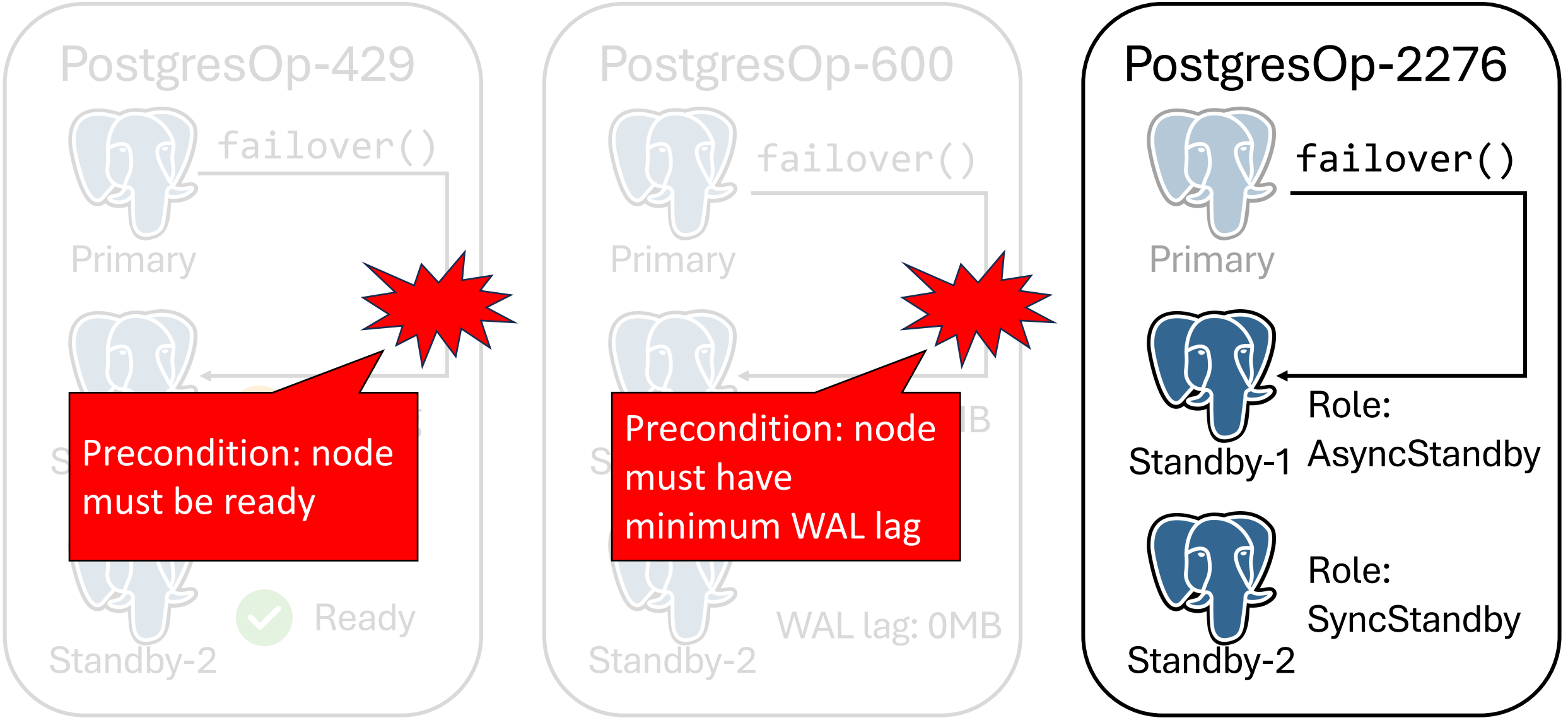
Preconditions are often unknown until violated



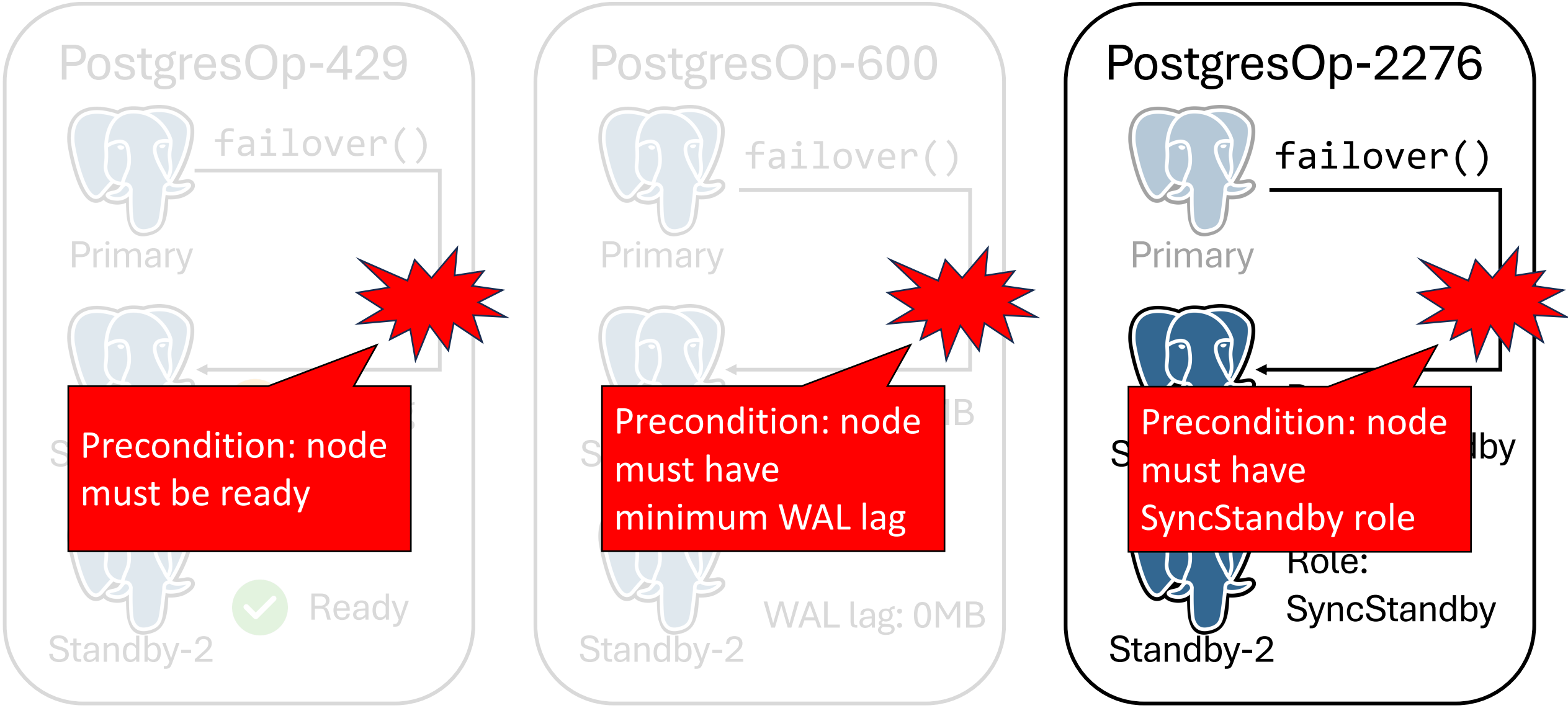
Preconditions are often unknown until violated



Preconditions are often unknown until violated

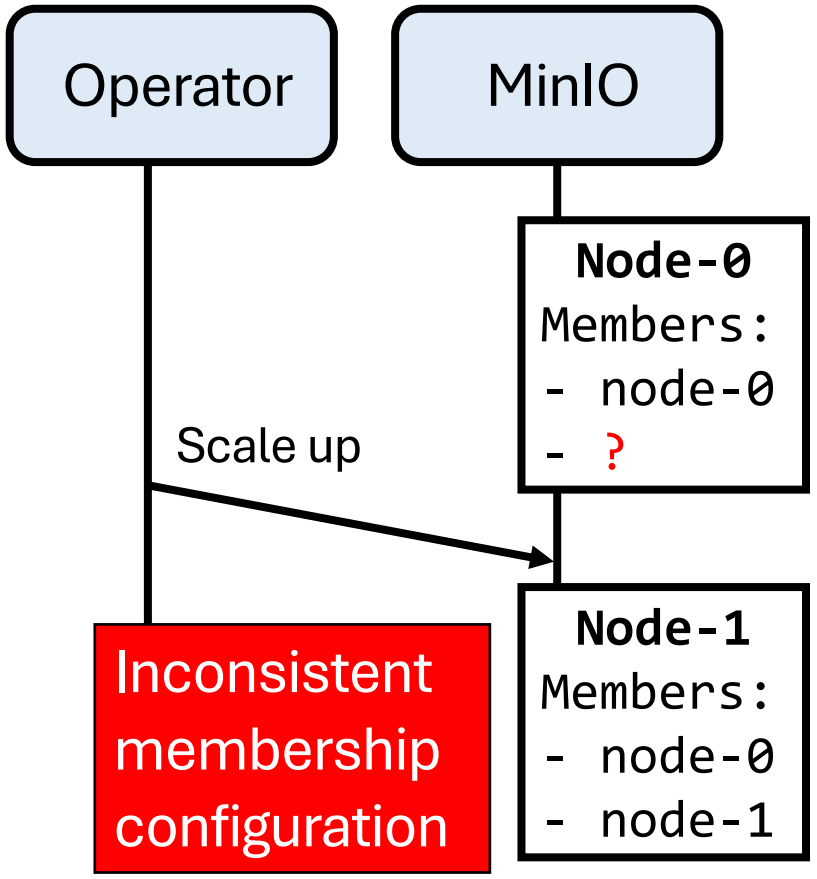


Preconditions are often unknown until violated



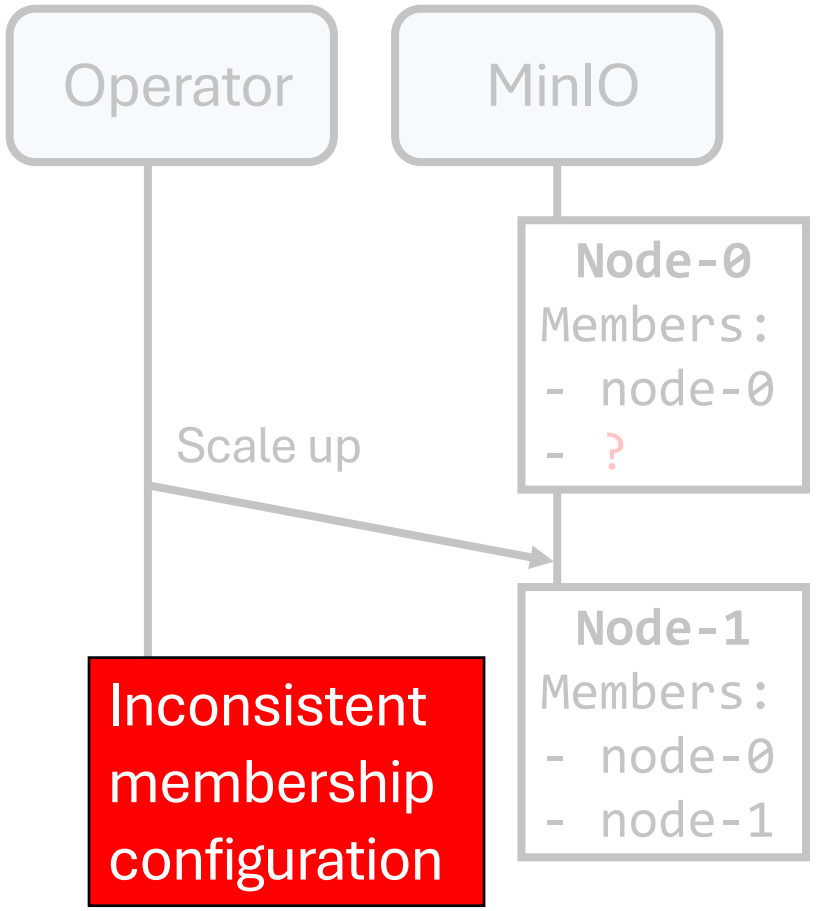
Semantic violations take many forms

Configuration (36%)

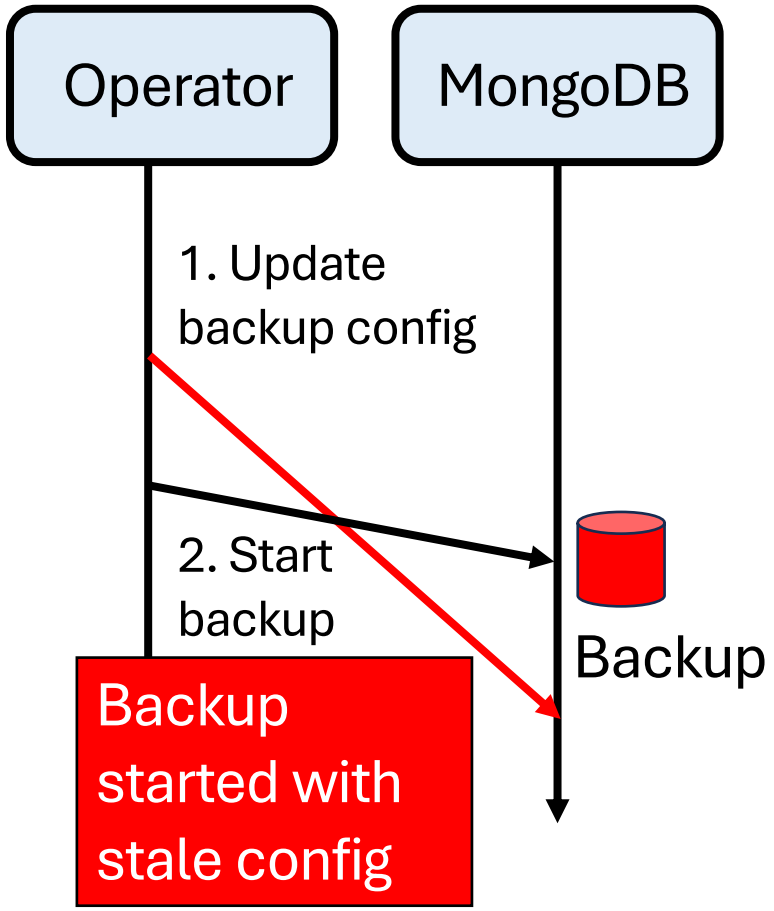


Semantic violations take many forms

Configuration (36%)

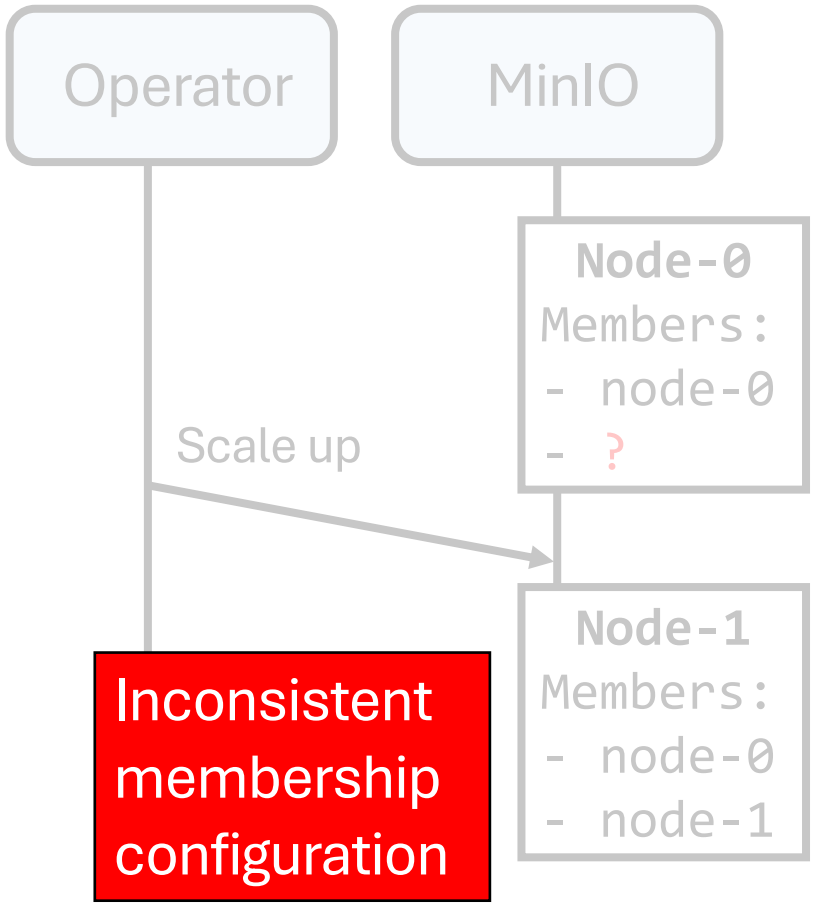


Ordering (31%)

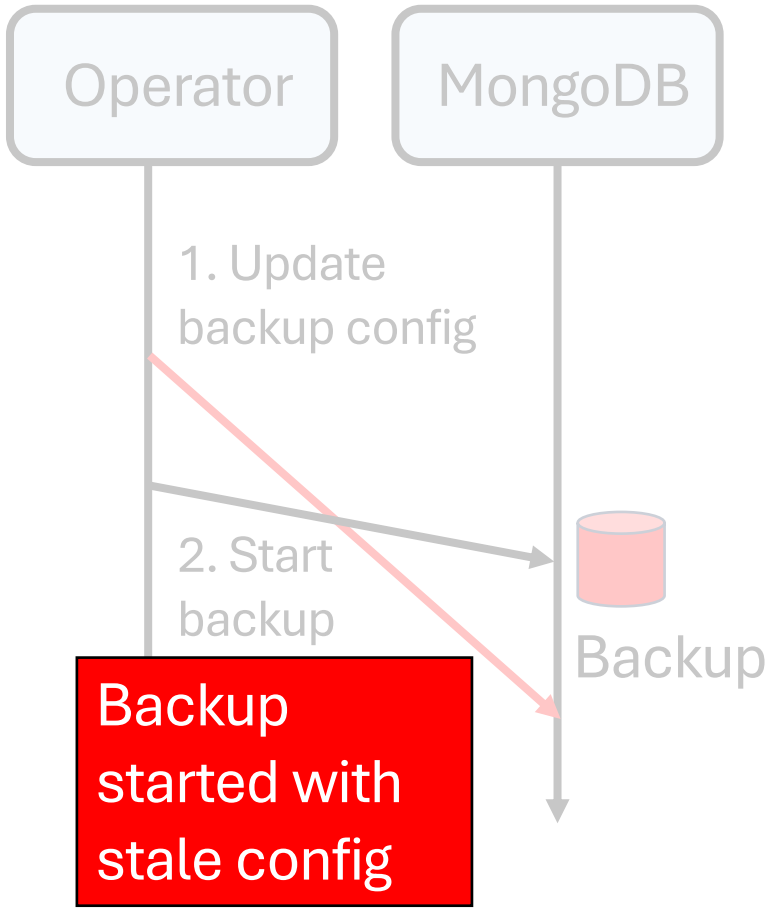


Semantic violations take many forms

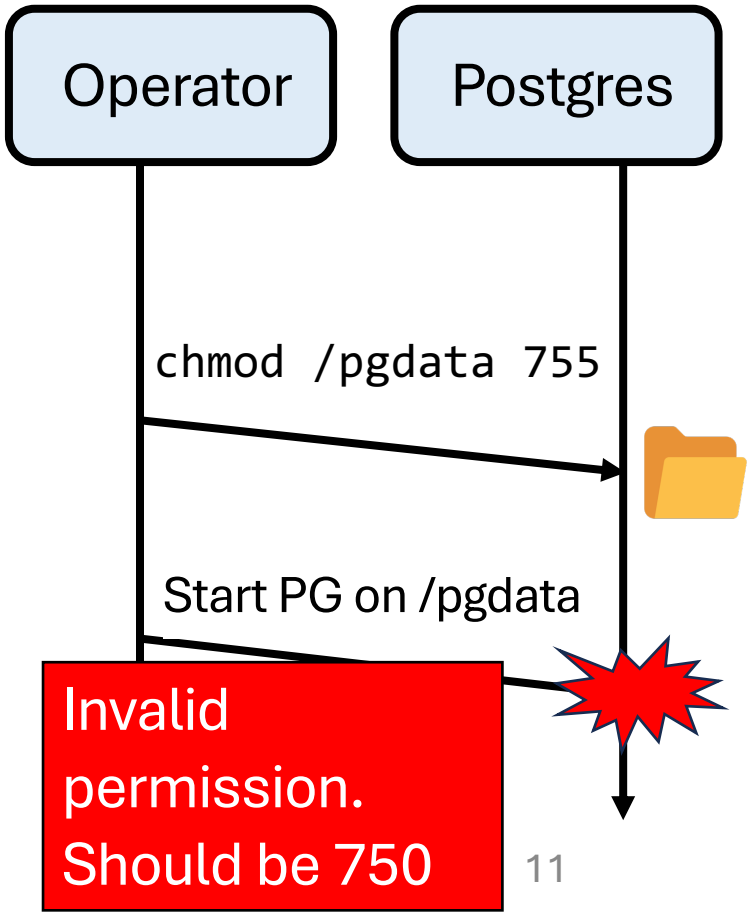
Configuration (36%)



Ordering (31%)



Environment (14%)



The lack of a formal management interface

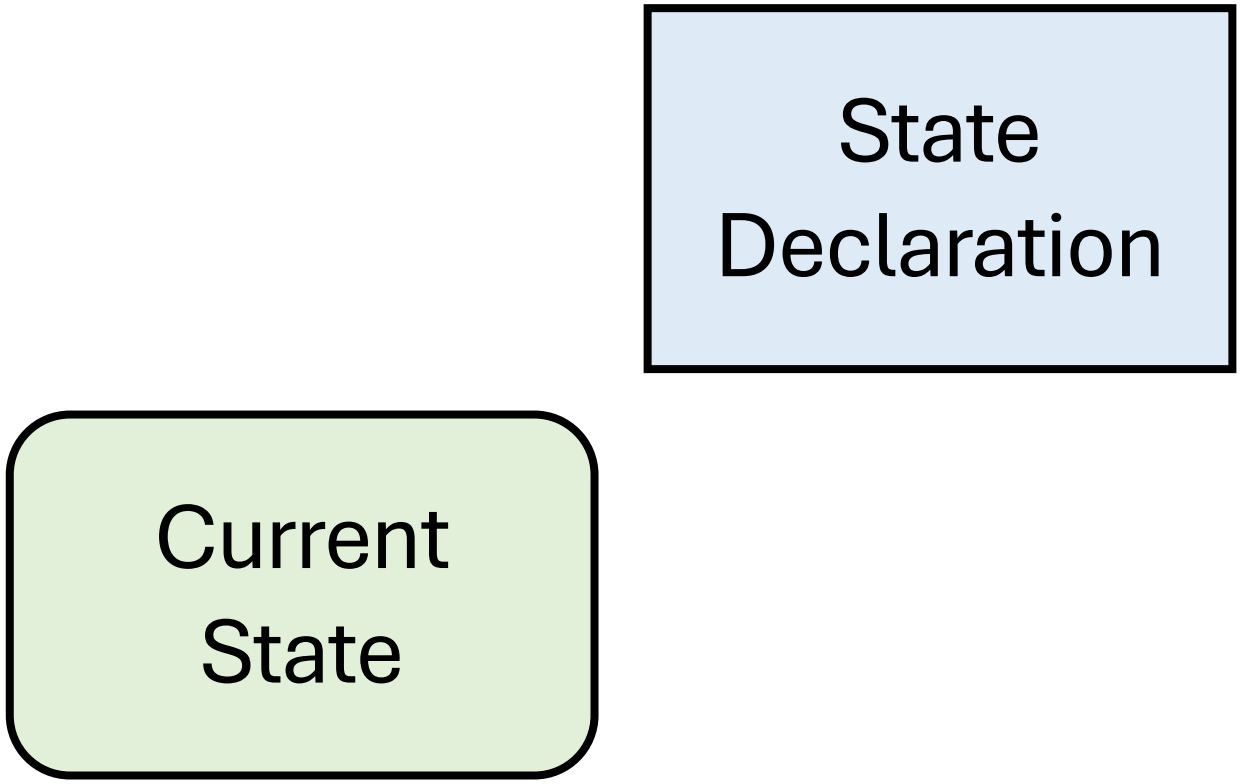
- No formal specification of operation semantics: configuration, preconditions, ordering, and environment

- Operators must reverse-engineer them from documentation, source code, and prior experience

Oat, a way to test against this gap

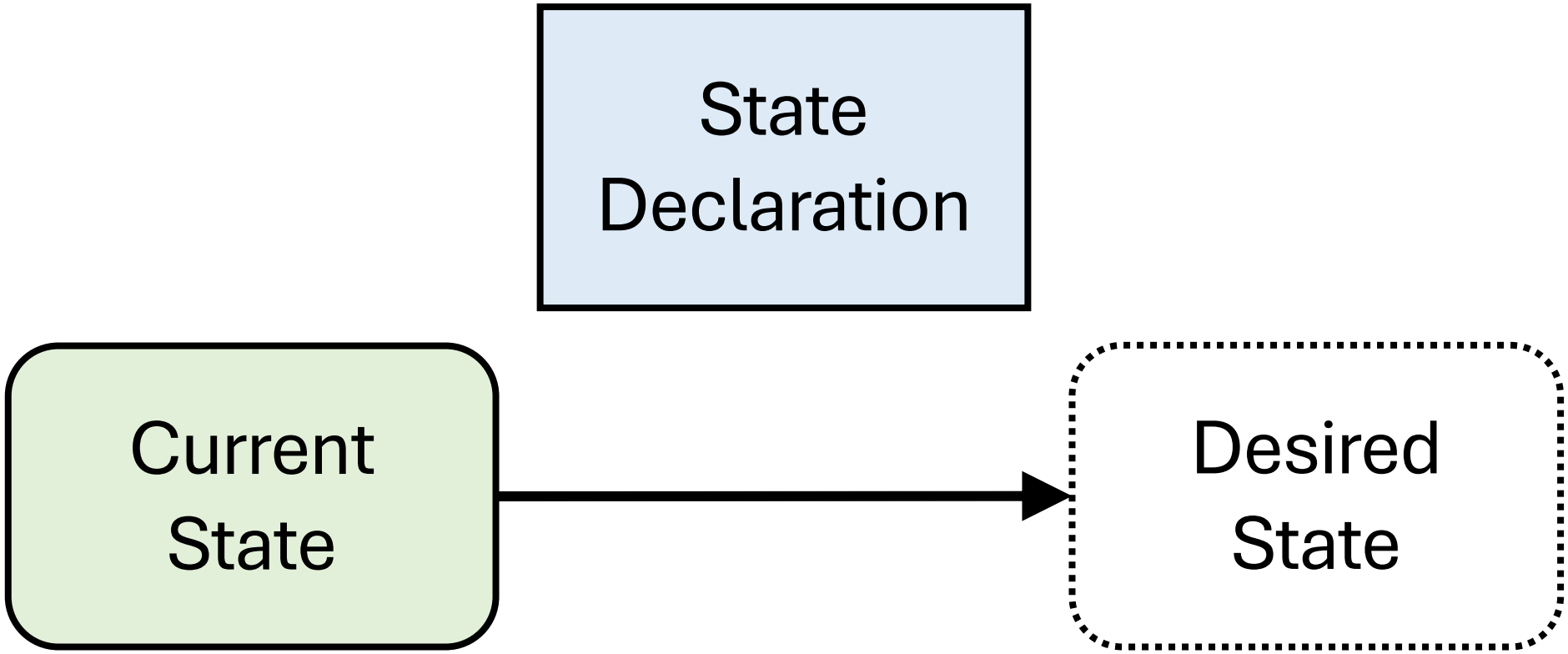
Oat tests operator-application interactions

- Oat follows the state-centric operator testing approach



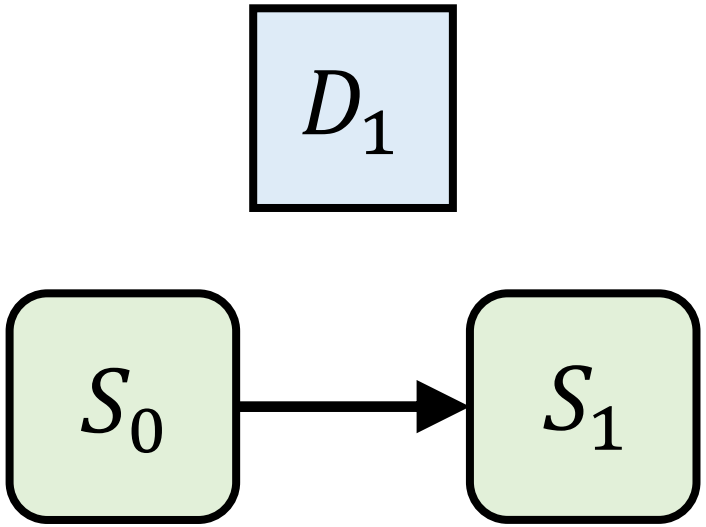
Oat tests operator-application interactions

- Oat follows the state-centric operator testing approach



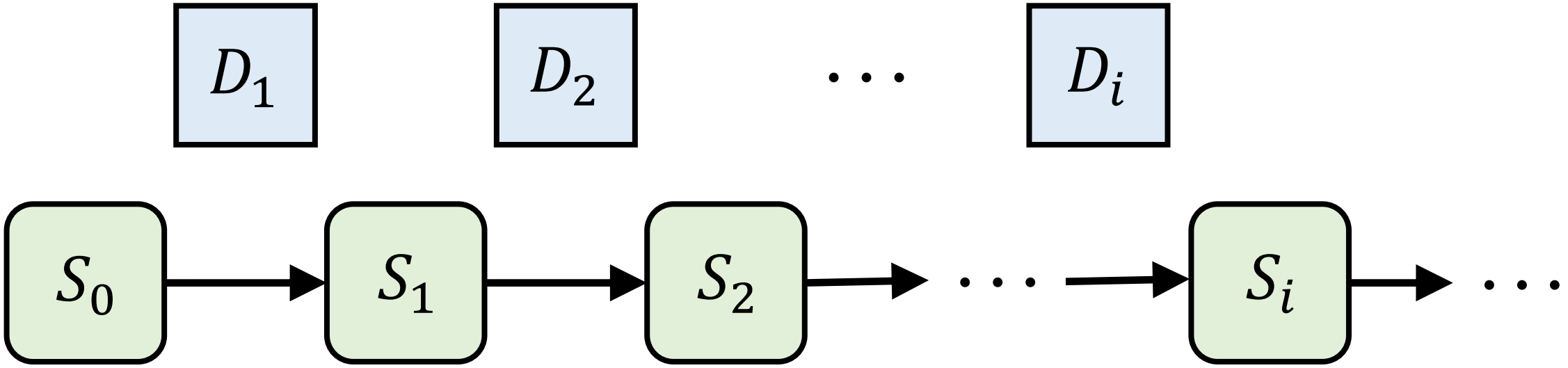
Oat tests operator-application interactions

- Oat follows the state-centric operator testing approach



Oat tests operator-application interactions

- Oat follows the state-centric operator testing approach



The study calls out two requirements

Findings

72% of failures triggered by state declarations need application-specific properties

Oat design

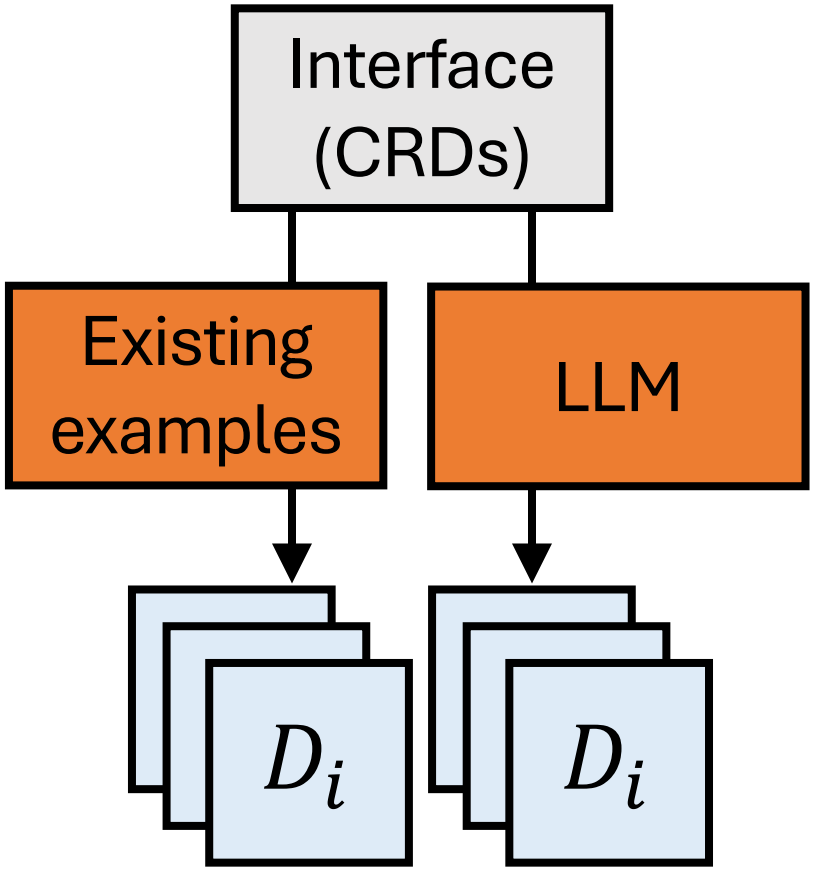
Synthesize app-specific state declarations

The study calls out two requirements

Findings	Oat design
 72% of failures triggered by state declarations need application-specific properties	Synthesize app-specific state declarations
 41% of failures need faults on the application	Inject app-targeted faults

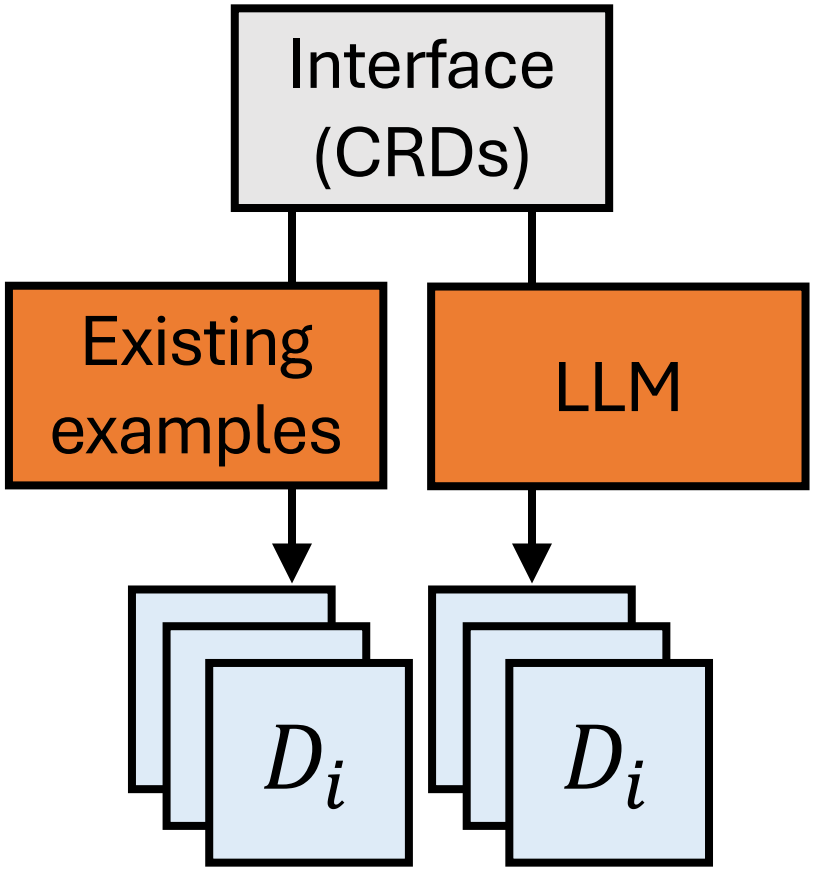
How Oat works

1. Synthesize app-specific declarations

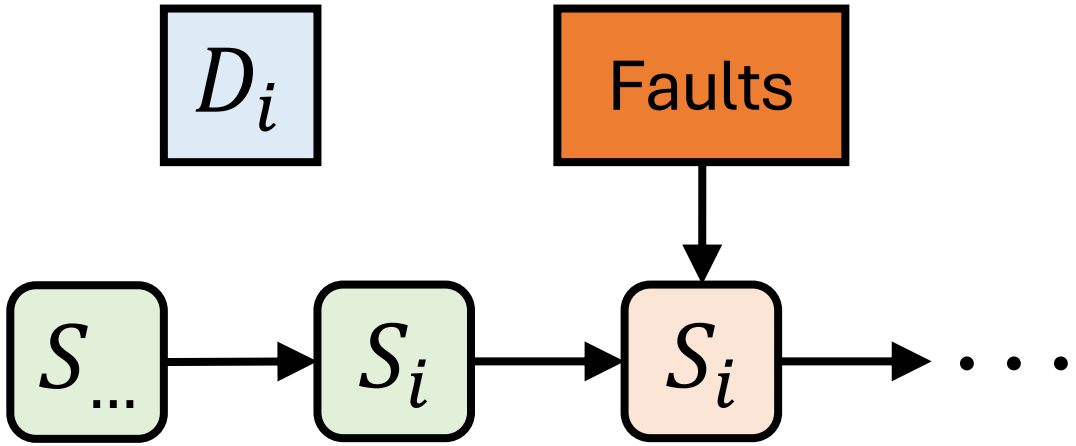


How Oat works

1. Synthesize app-specific declarations

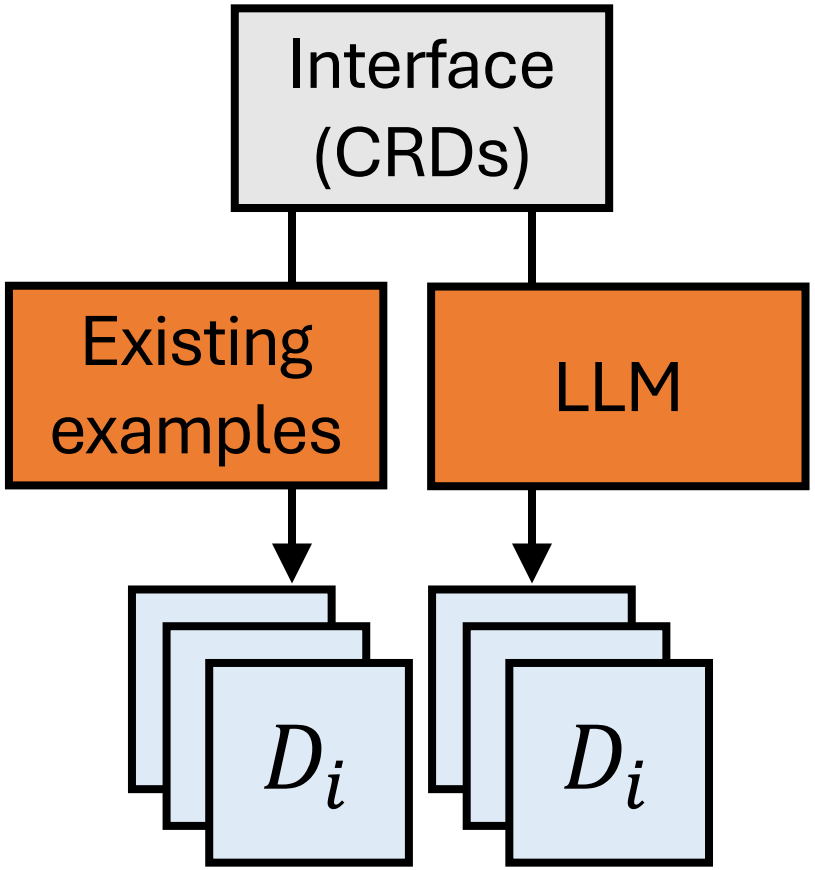


2. Execute test campaign

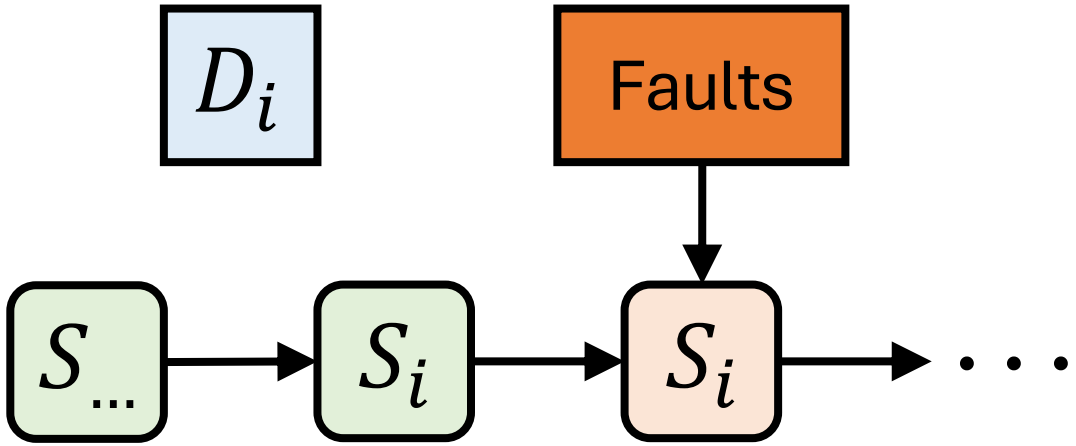


How Oat works

1. Synthesize app-specific declarations



2. Execute test campaign



3. Automatic oracles to flag bugs

Bug Reports

Oat found 86 new bugs across 6 mature operators

Operator	Operation Semantics	State Observability	Version Compatibility	Error Handling	Other	Total
CassOp	7	0	1	0	2	10
KafkaOp	2	1	0	0	0	3
MariaDBOp	9	1	0	1	16	27
MinIOOp	1	0	0	0	1	2
MongoOp	18	2	1	3	2	26
TiDBOp	9	2	1	0	6	18
Total	**46**	**6**	**3**	**4**	**27**	**86**

Oat found 86 new bugs across 6 mature operators

Operator	Operation Semantics	State Observability	Version Compatibility	Error Handling	Other	Total
CassOp	7	0	1	0	2	10
KafkaOp	2	1	0	0	0	3
MariaDBOp	9	1	0	1	16	27
MinIOOp	1	0	0	0	1	2
MongoOp	18	2	1	3	2	26
TiDBOp	9	2	1	0	6	18
Total	**46**	**6**	**3**	**4**	**27**	**86**

Oat found 86 new bugs across 6 mature operators

Operator	Operation Semantics	State Observability	Version Compatibility	Error Handling	Other	Total
CassOp	7	0	1	0	2	10
KafkaOp	2	1	0	0	0	3
MariaDBOp	9	1	0	1	16	27
MinIOOp	1	0	0	0	1	2
MongoOp	18	2	1	3	2	26
TiDBOp	9	2	1	0	6	18
Total	**46**	**6**	**3**	**4**	**27**	**86**

Manageability should be first-class design principle

- **An empirical study on real-world operator failures**
  - Operator-application interactions are the largest, most overlooked
- Cloud applications lack **well-specified management interfaces**
  - Operator-application interactions become error-prone
  - As **AI agents** replace rule-based operators, formal management interfaces become essential safety guardrails
- **Oat** as a practical solution to testing against this gap
  - Found **86 new bugs** in 6 mature operators