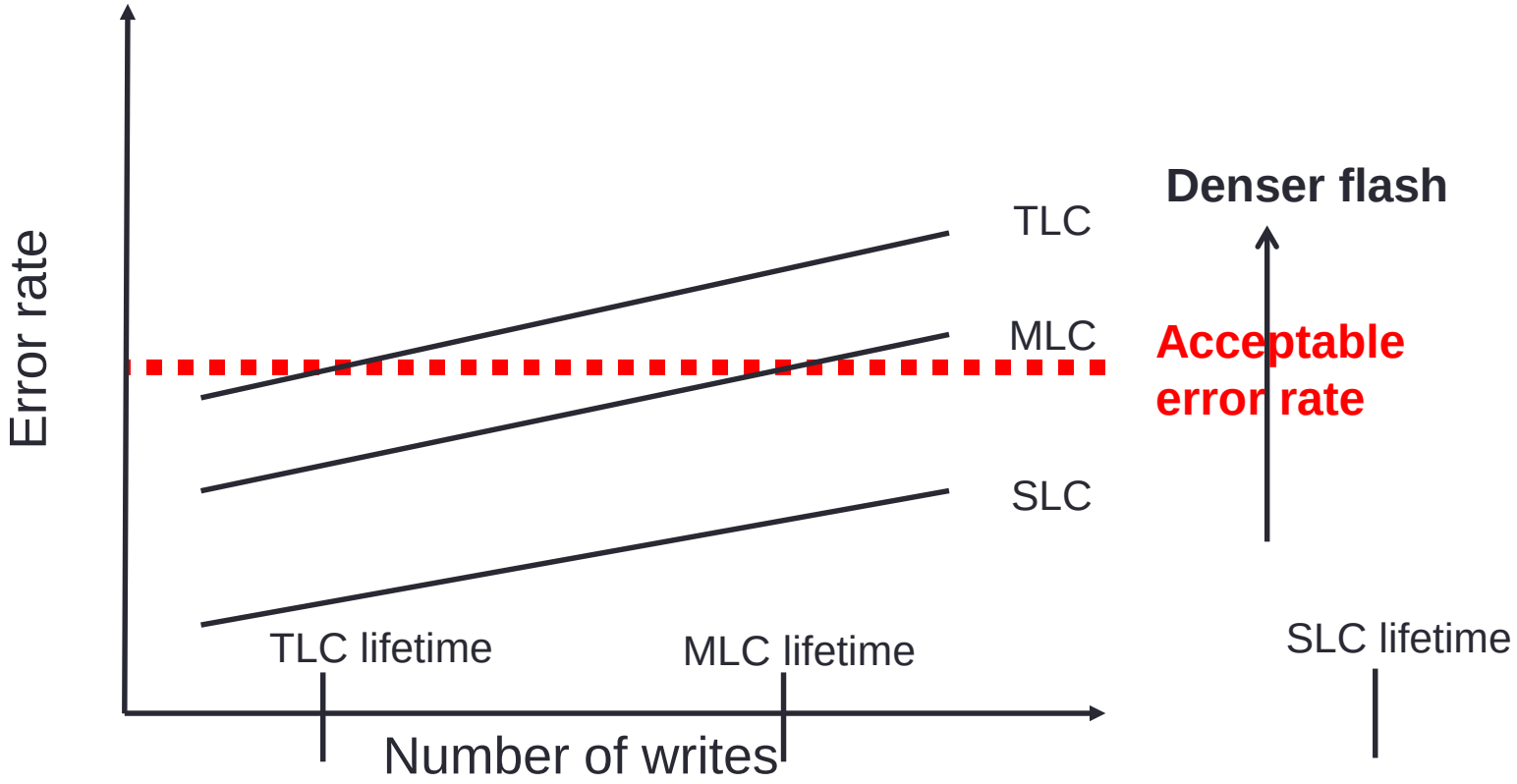
RETHINKING END-TO-END RELIABILITY IN CLOUD STORAGE SYSTEMS

Amy Tai, Andrew Kryczka, Shobhit Kanaujia, Kyle Jamieson, Michael J. Freedman, **Asaf Cidon**

To appear in Usenix ATC 2019

Denser flash → shorter lifetime



Source: Novotný, R., J. Kadlec, and R. Kuchta. "NAND Flash Memory Organization and Operations." Journal of Information Technology & Software Engineering 5.1 (2015): 1.

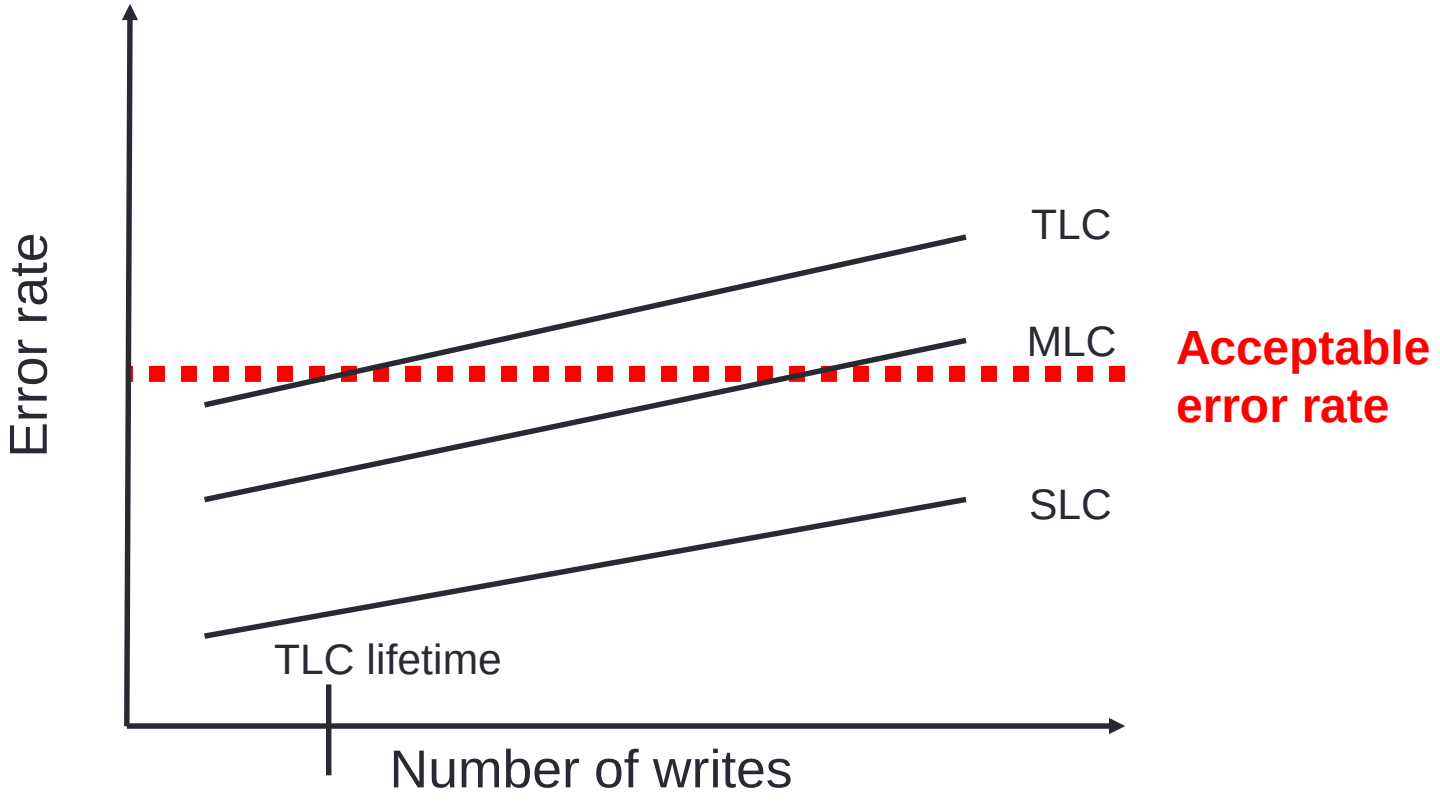
Shorter flash lifetimes are a problem

- Datacenter operators must closely monitor flash writes
- Memor
  - →
  - →

How can we increase flash lifetimes?

- Datacenters struggling to adopt future generations of flash (e.g., QLC)

Increasing acceptable error rate → increase lifetimes



Source: Novotný, R., J. Kadlec, and R. Kuchta. "NAND Flash Memory Organization and Operations." Journal of Information Technology & Software Engineering 5.1 (2015): 1.

But.. hardware is expected to have low error rates

- Software is designed so bit errors are rare
  - Bit errors errors cause failed operations and reduced availability
  - Error-handling path is not performant

Distributed error Isolation and RECovery Techniques (DIRECT)

1. Use distributed redundancy to fix local bit errors
   - Distributed systems need redundant copies for availability
2. Optimize error-recovery performance

→

flash devices can expose high error rates

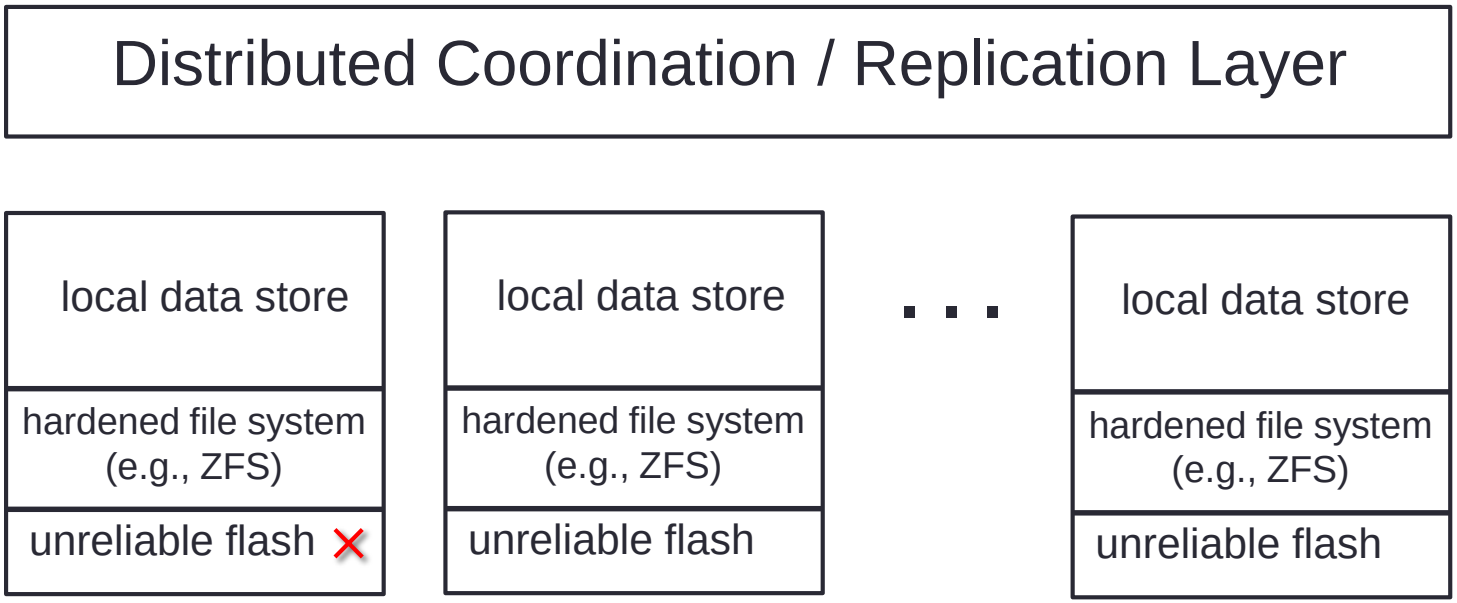
→

flash devices have longer lifetimes

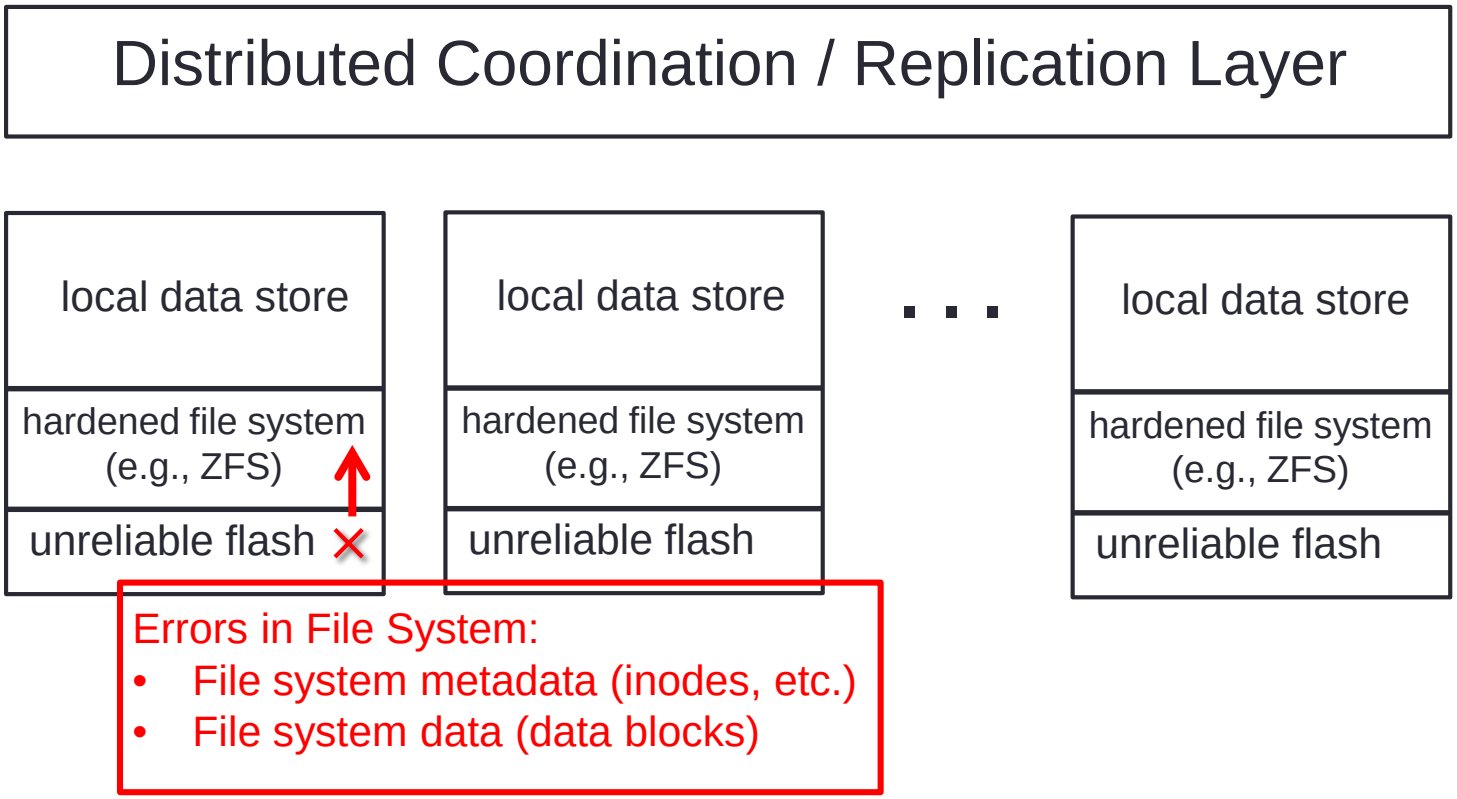
→

cheaper flash devices (QLC and beyond)

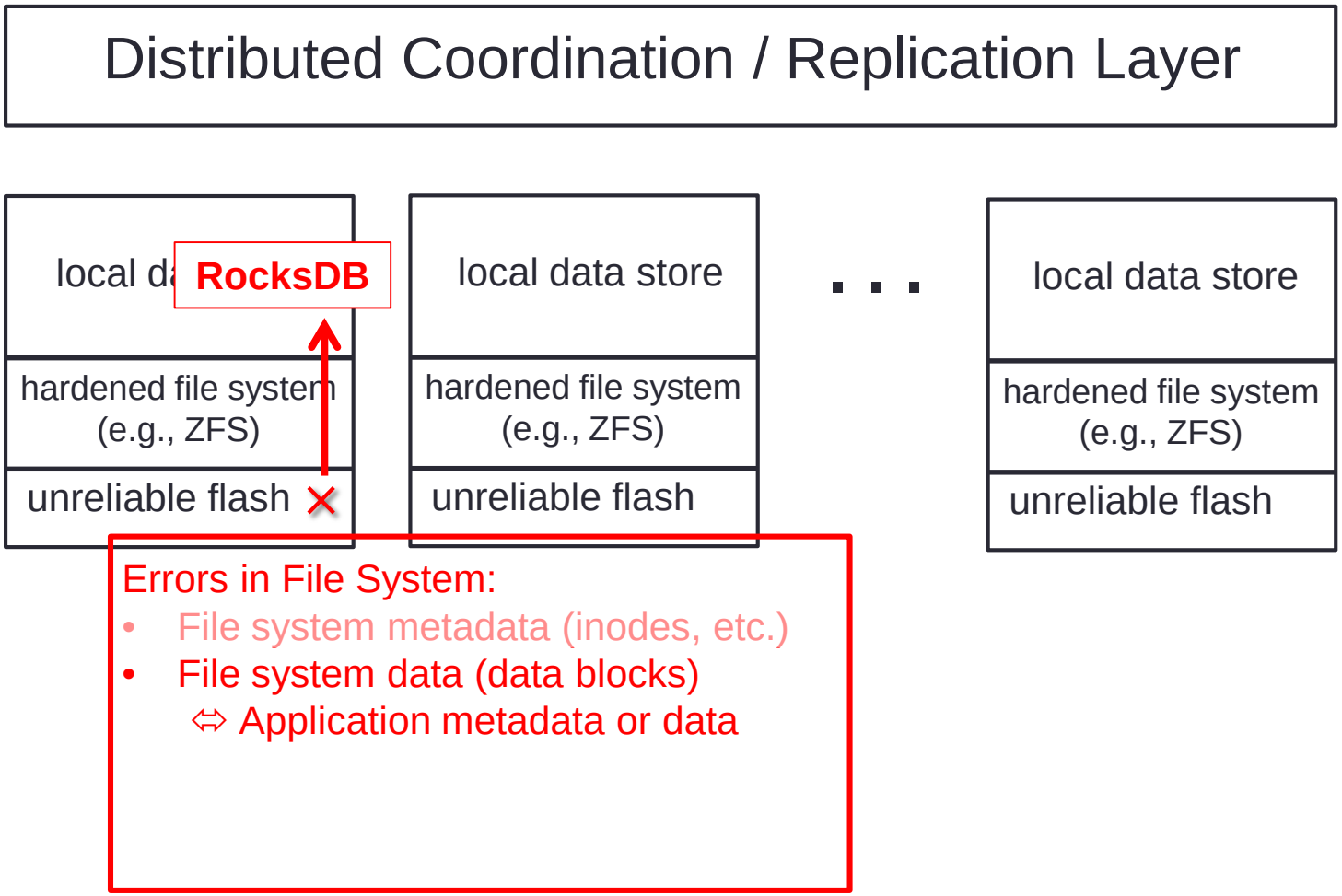
Bit errors in the storage stack…



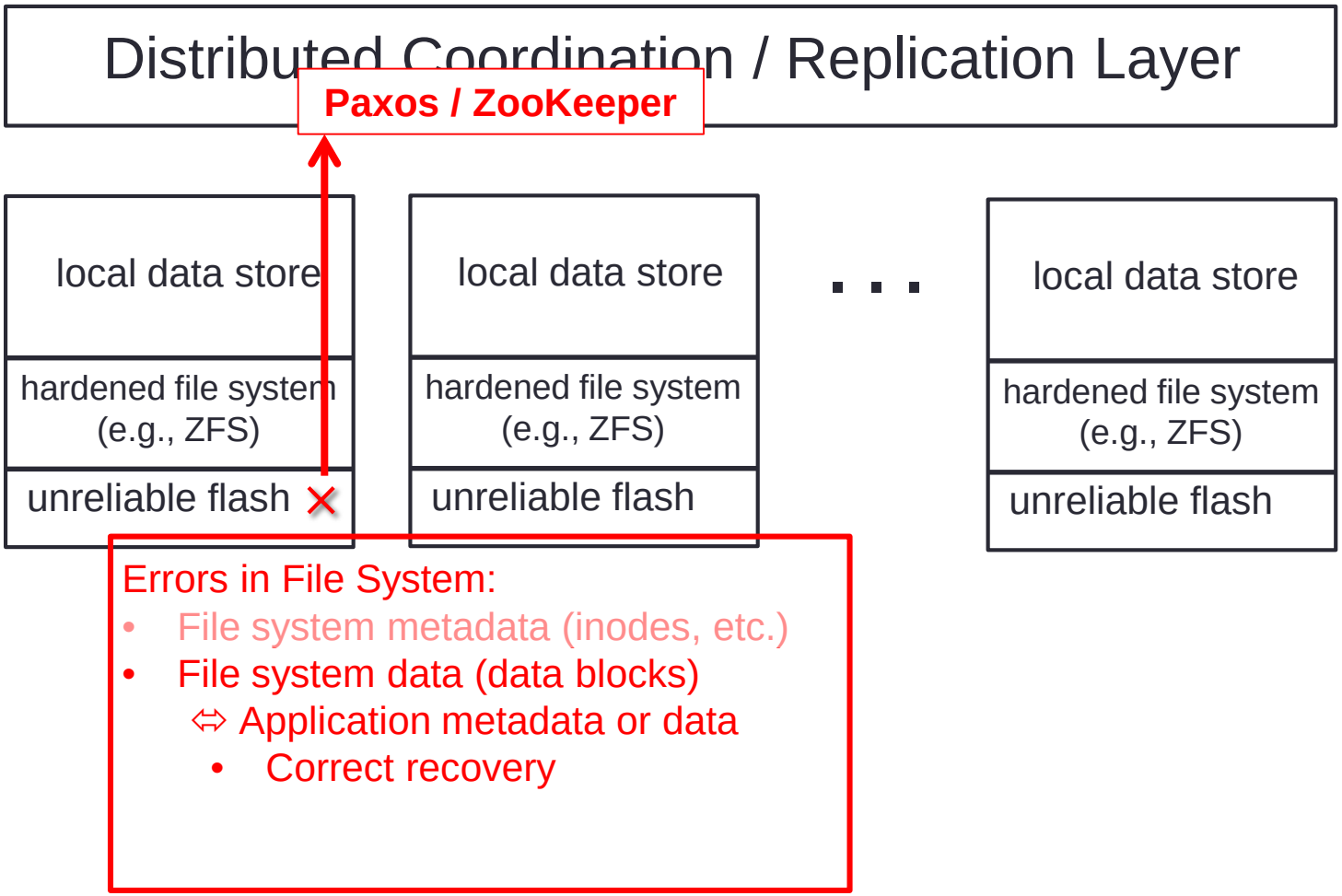
… can manifest in the file system



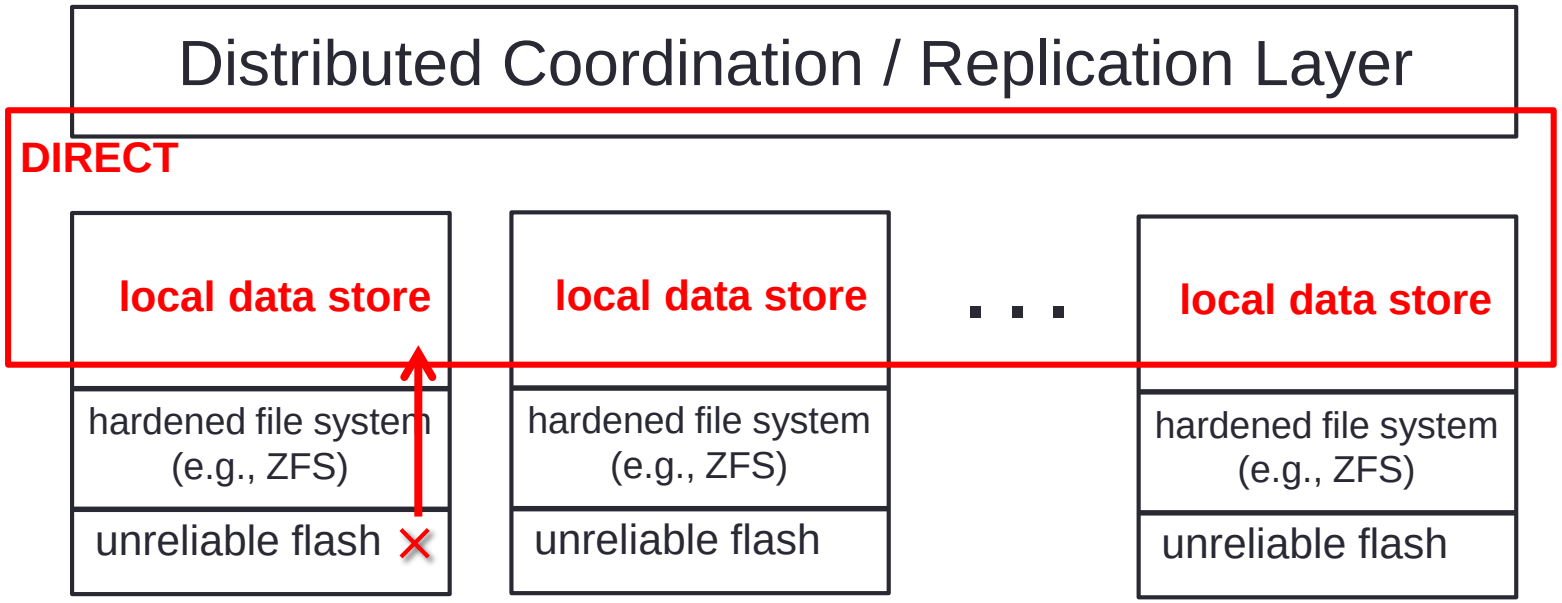
…or in the local data store



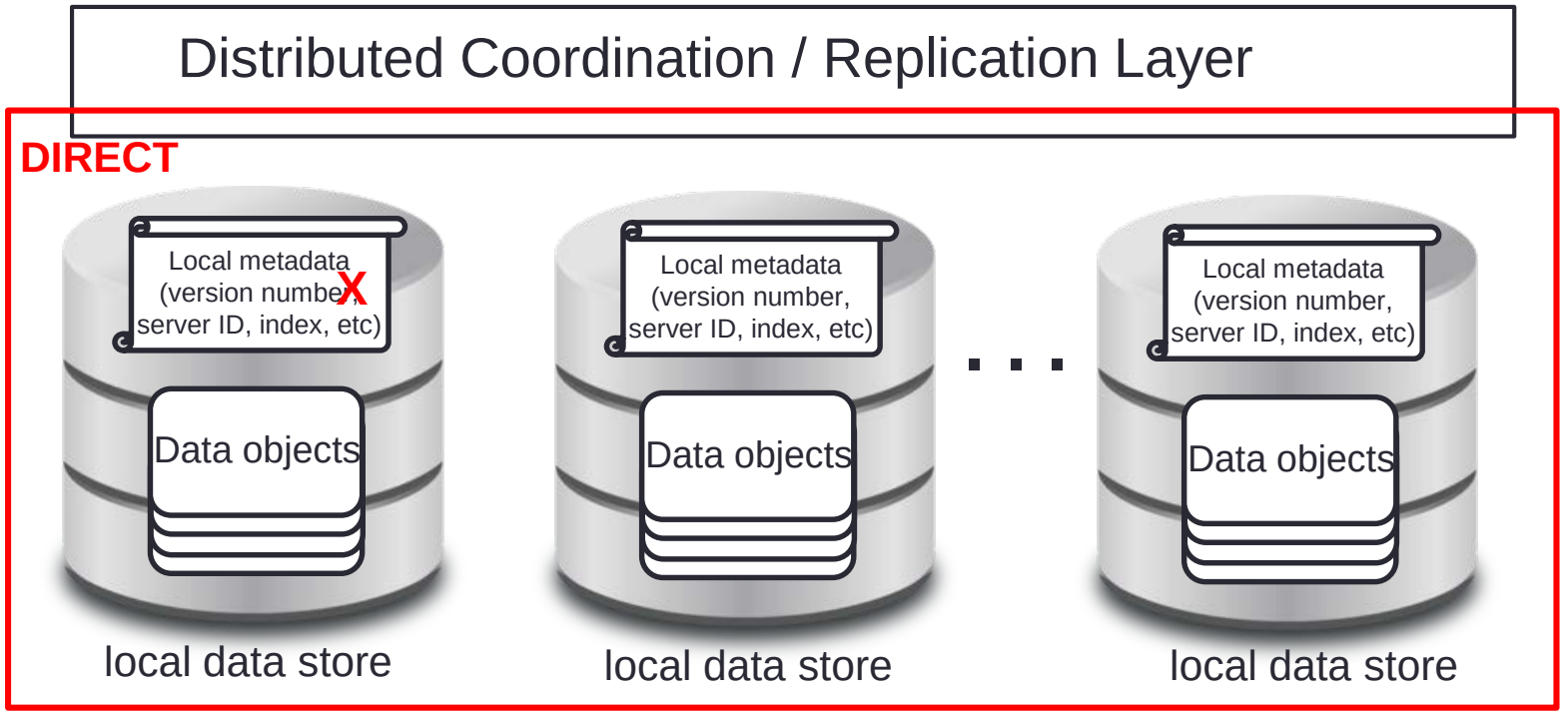
…and need to be dealt with in the coordination layer



DIRECT corrects bit errors in the local data store



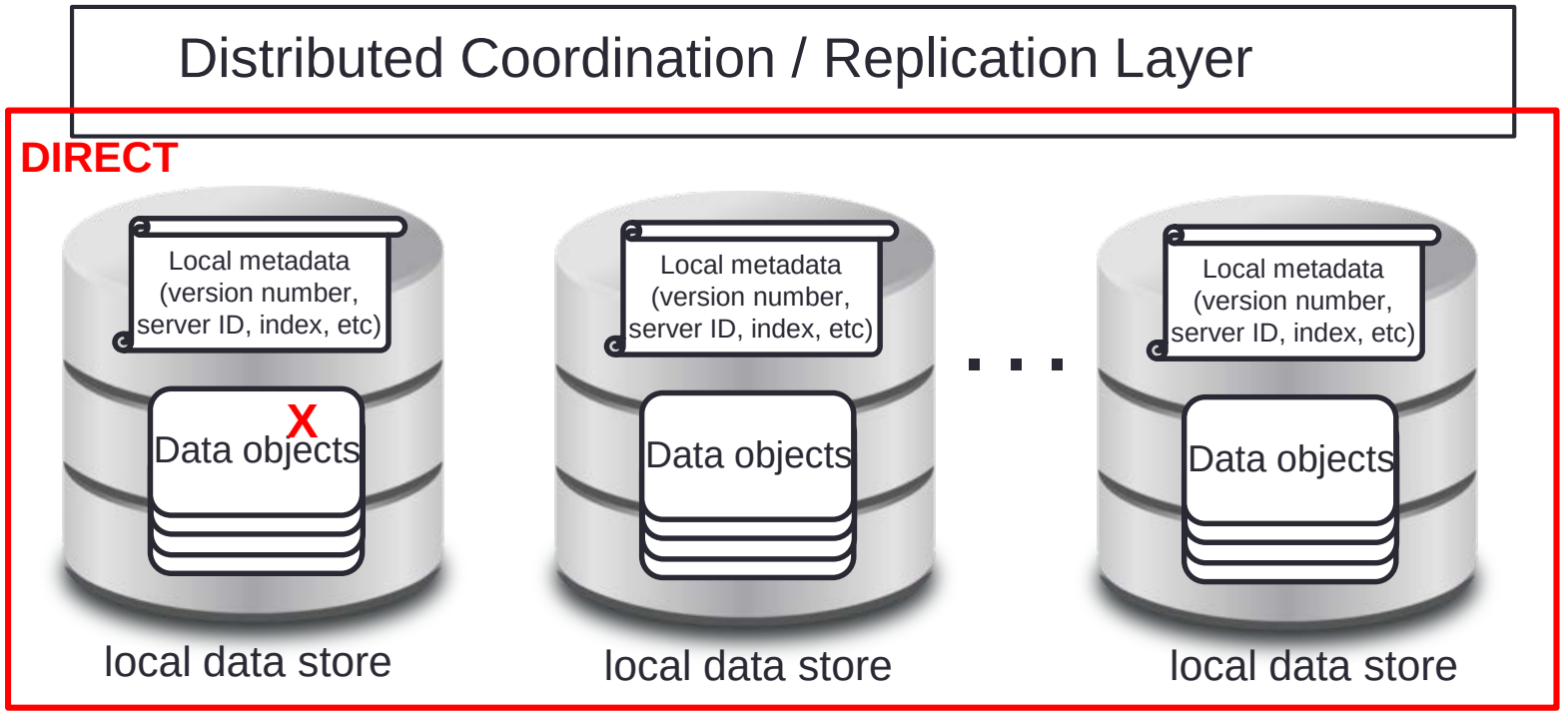
Local data store errors: metadata



DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata

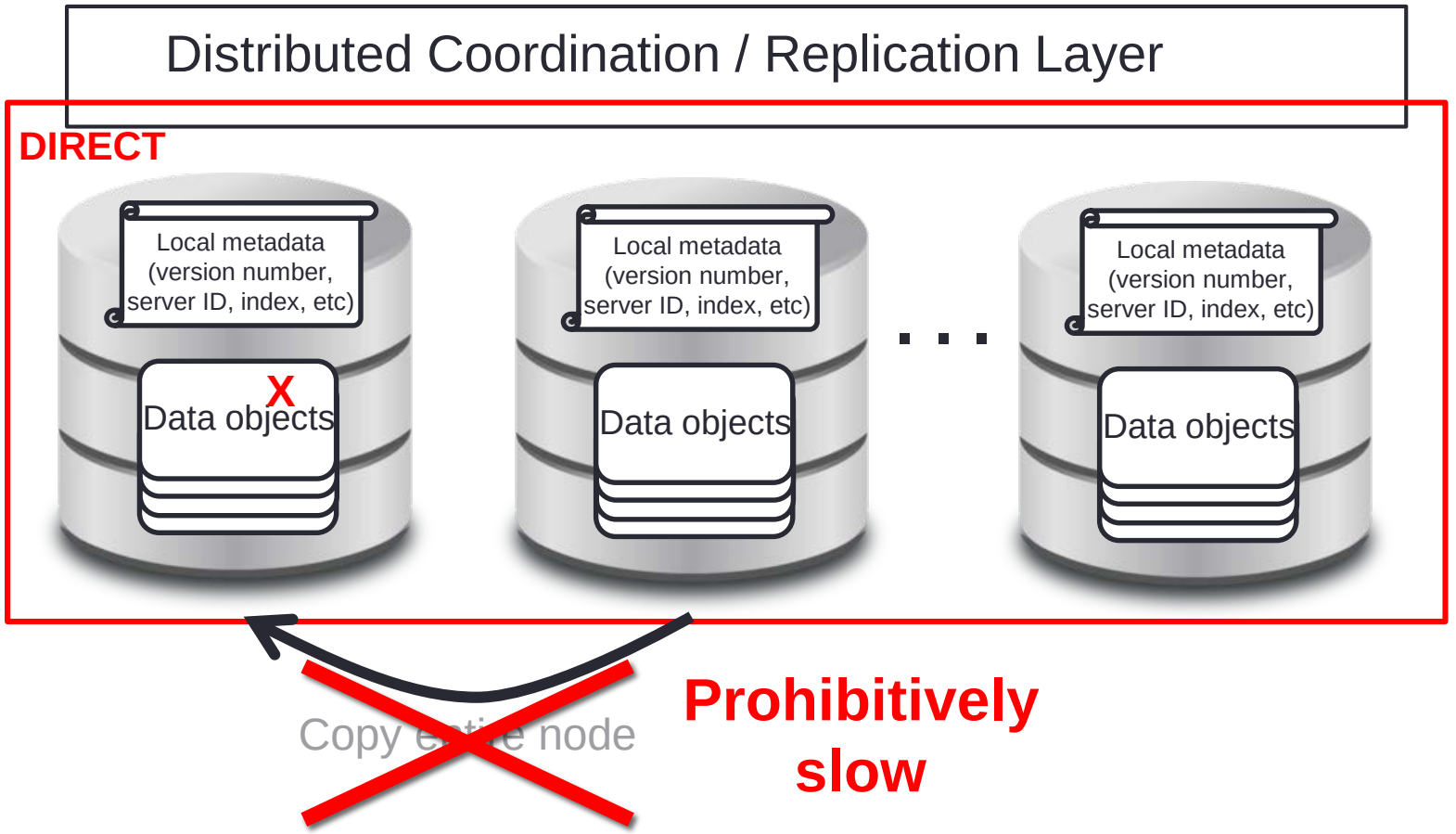
Local data store errors: data



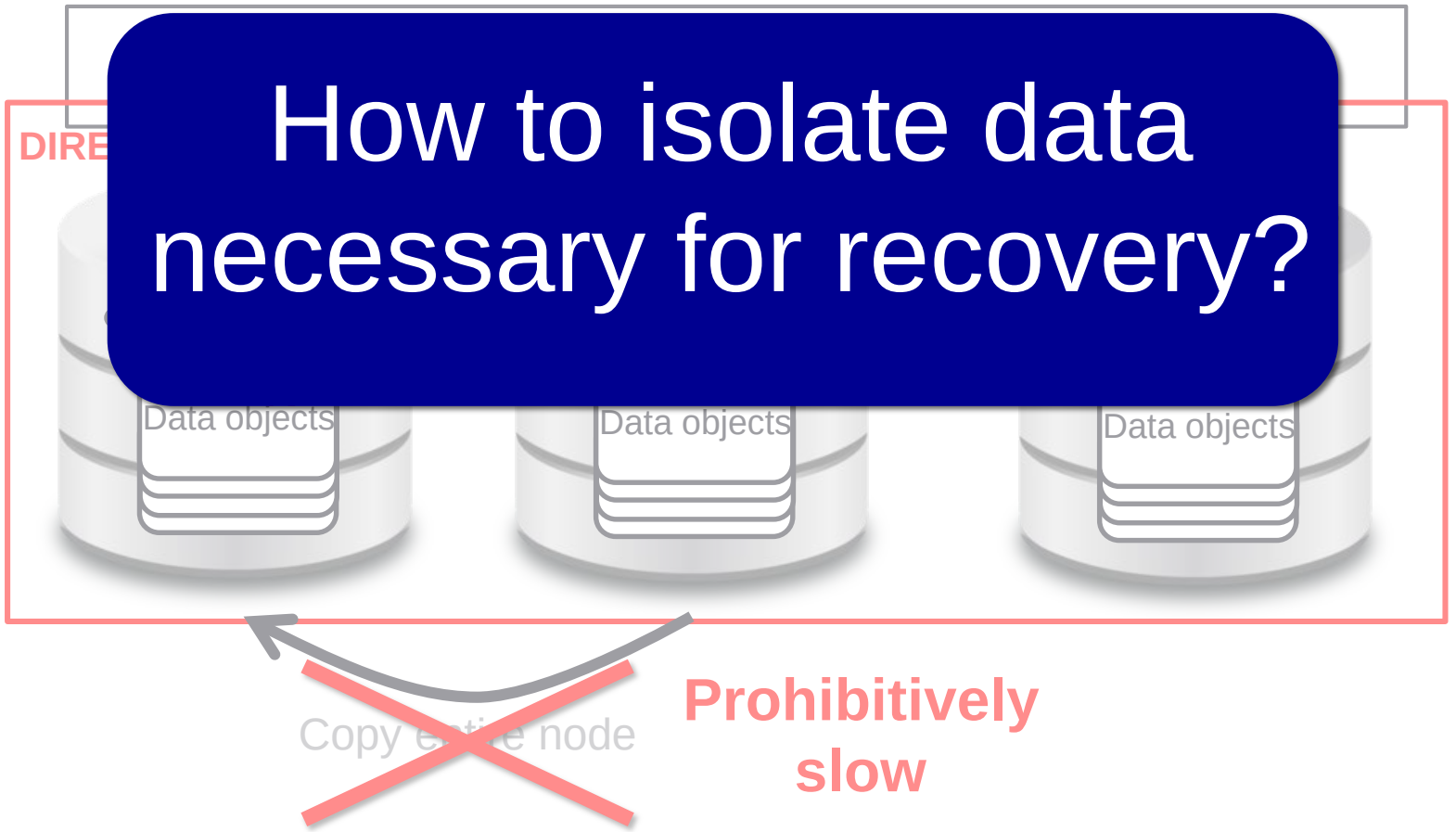
DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas

Optimizing error recovery: strawman treats bit errors as unavailability events



Optimizing error recovery: strawman treats bit errors as unavailability events



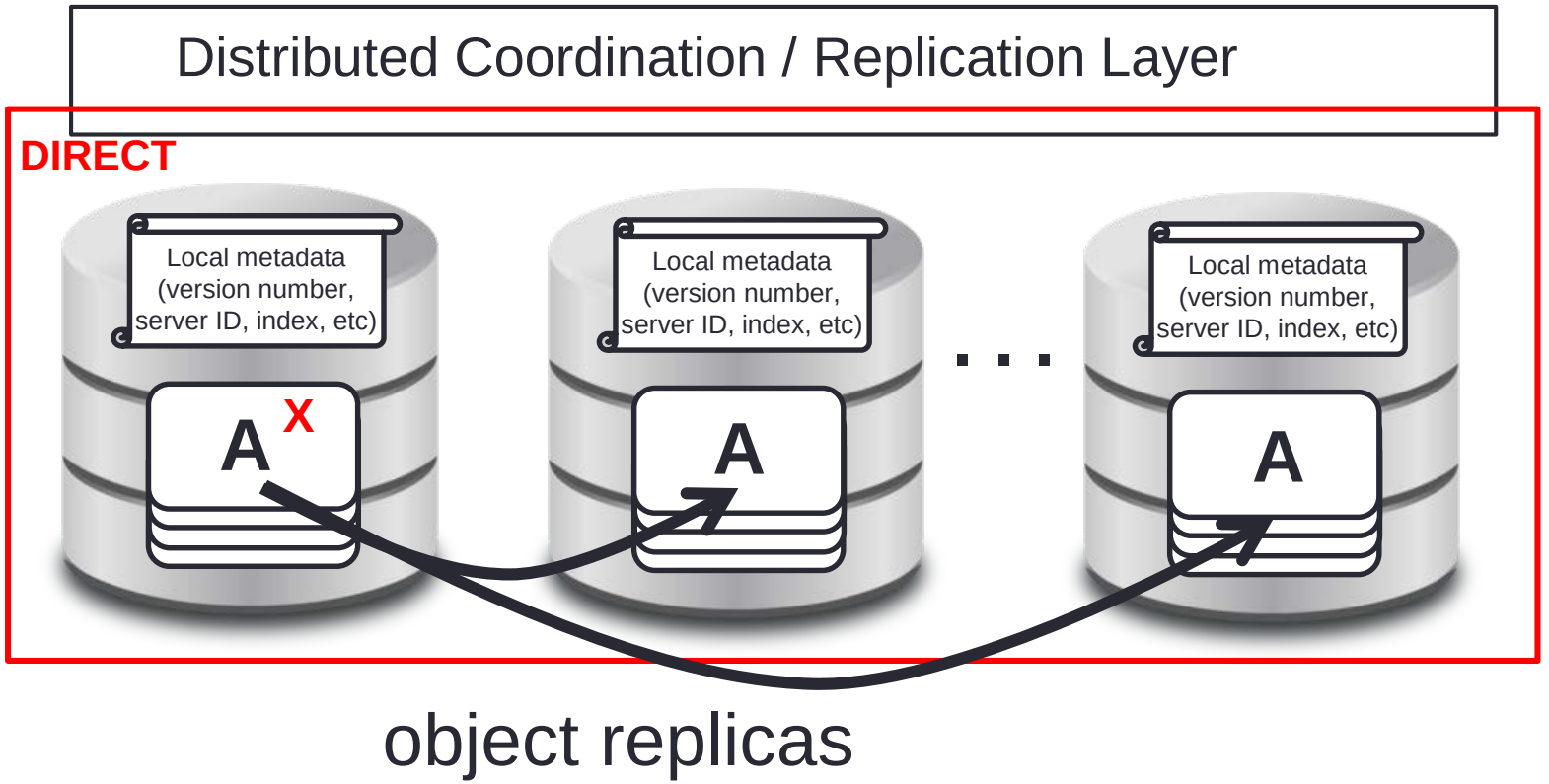
DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems

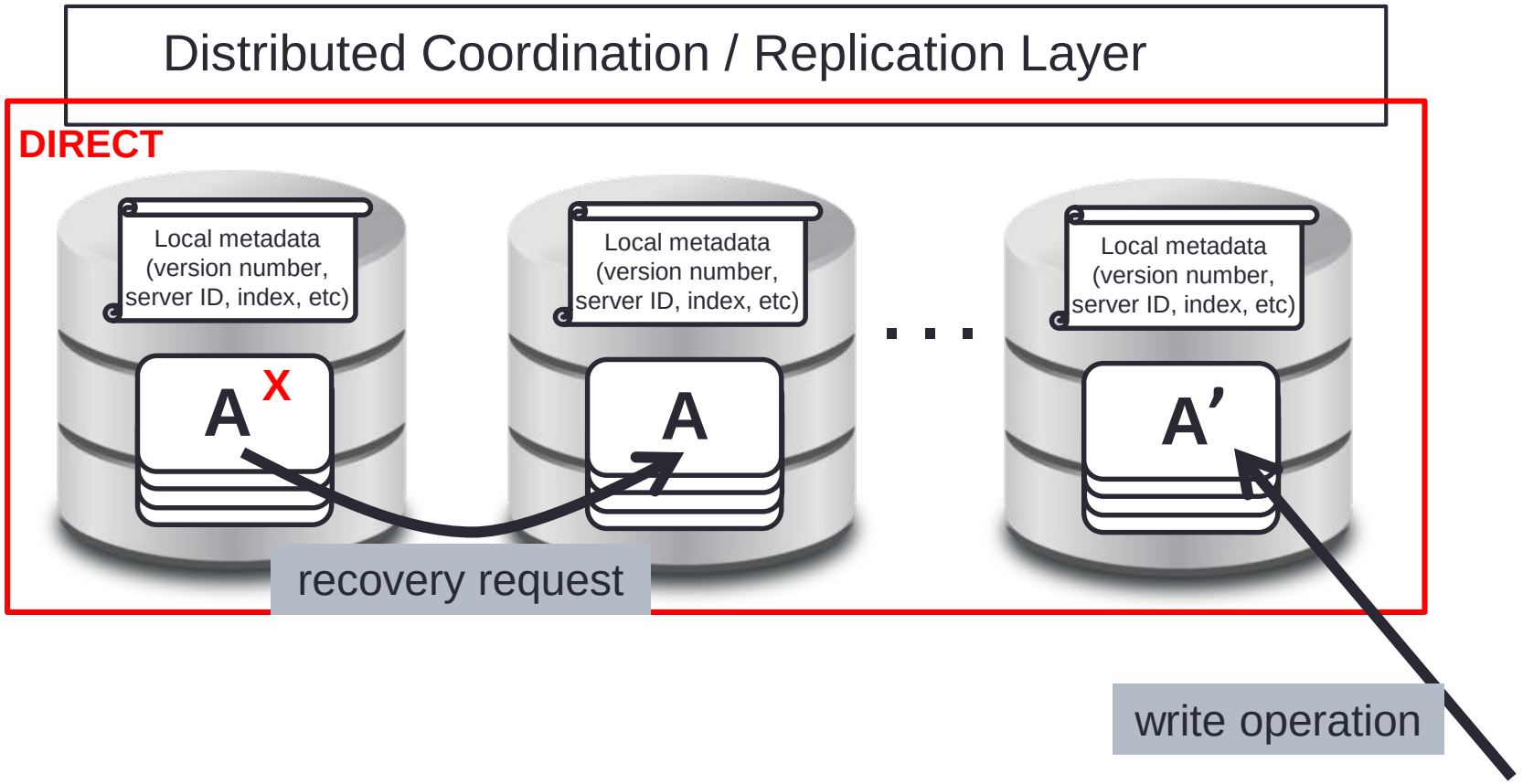
DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems
3. Safe recovery

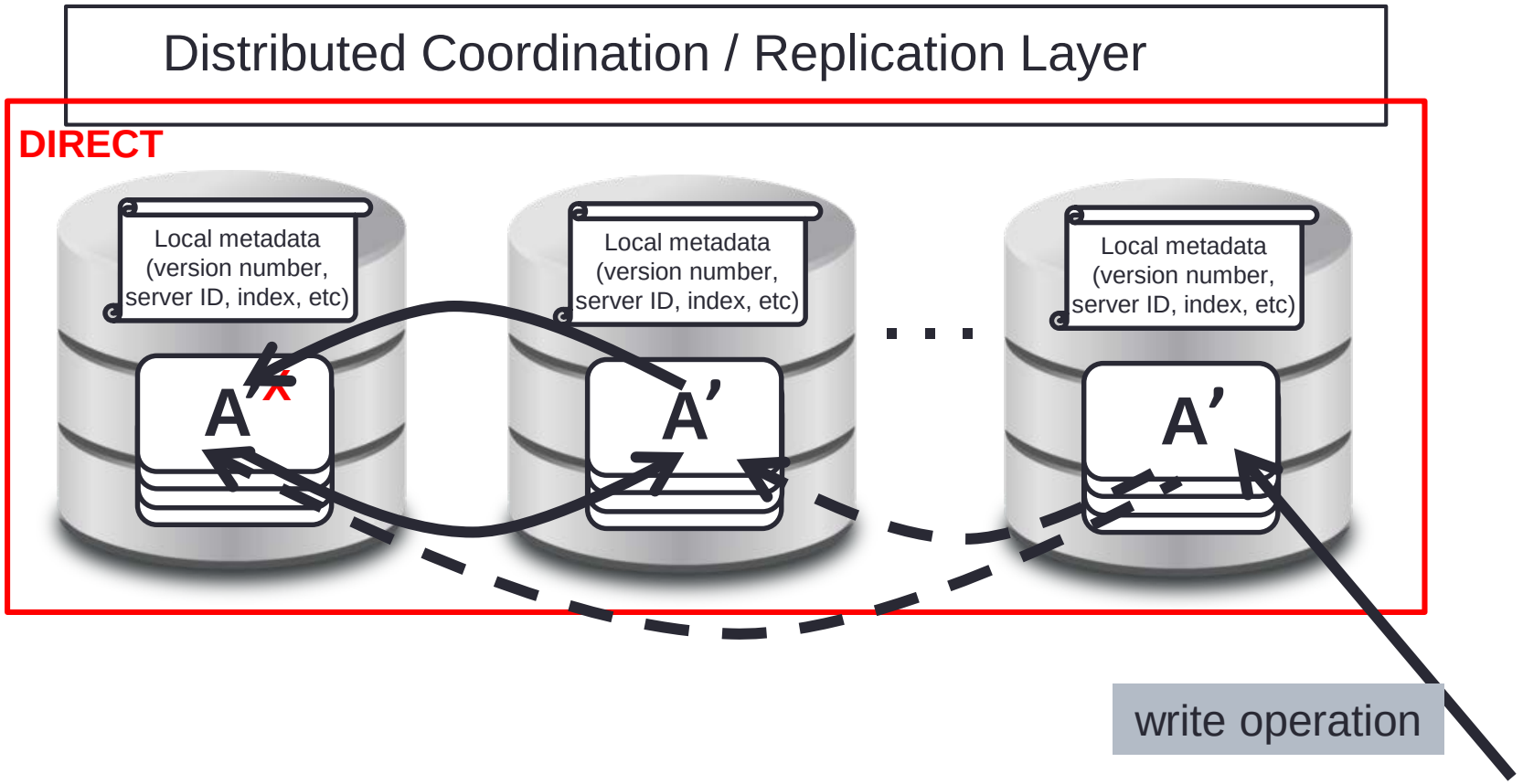
Naïve recovery protocol



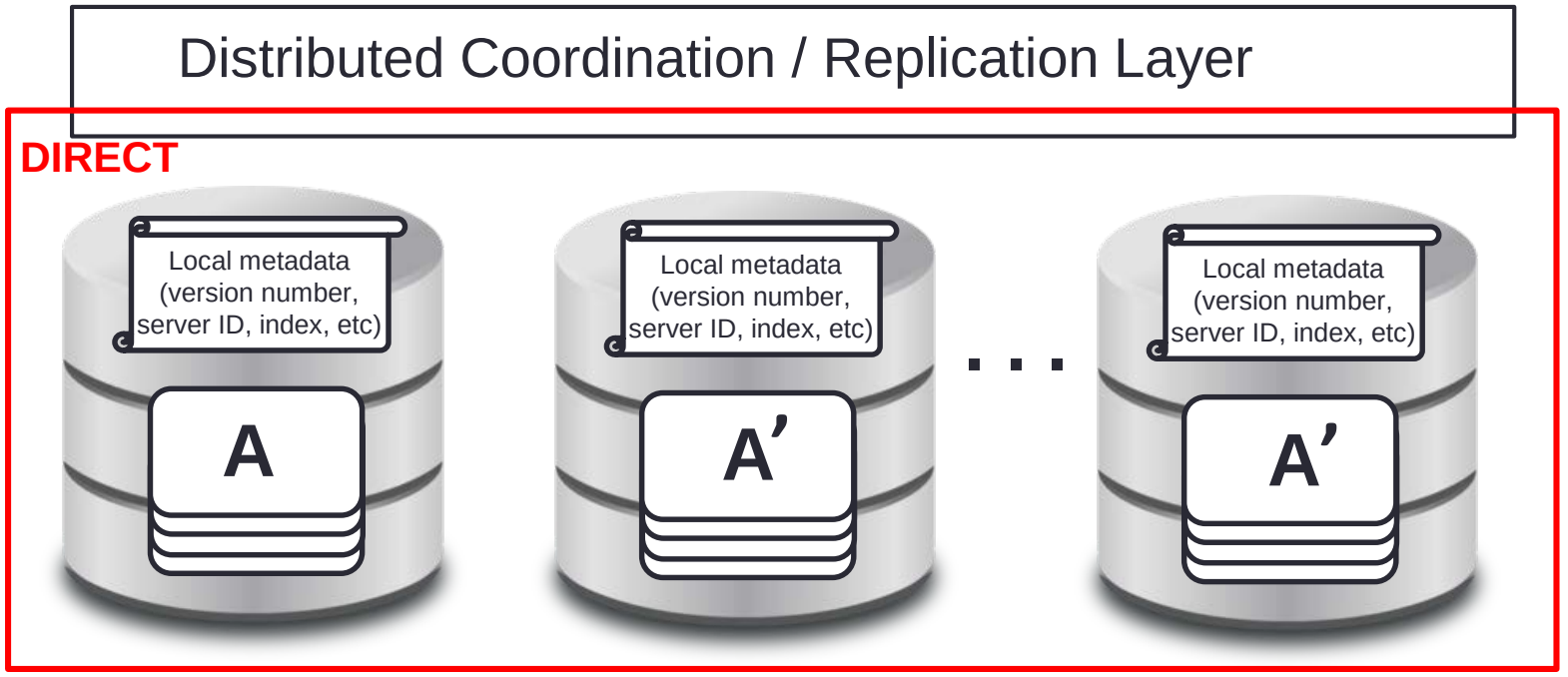
Naïve recovery protocol



Naïve recovery protocol



Naïve recovery protocol: inconsistency



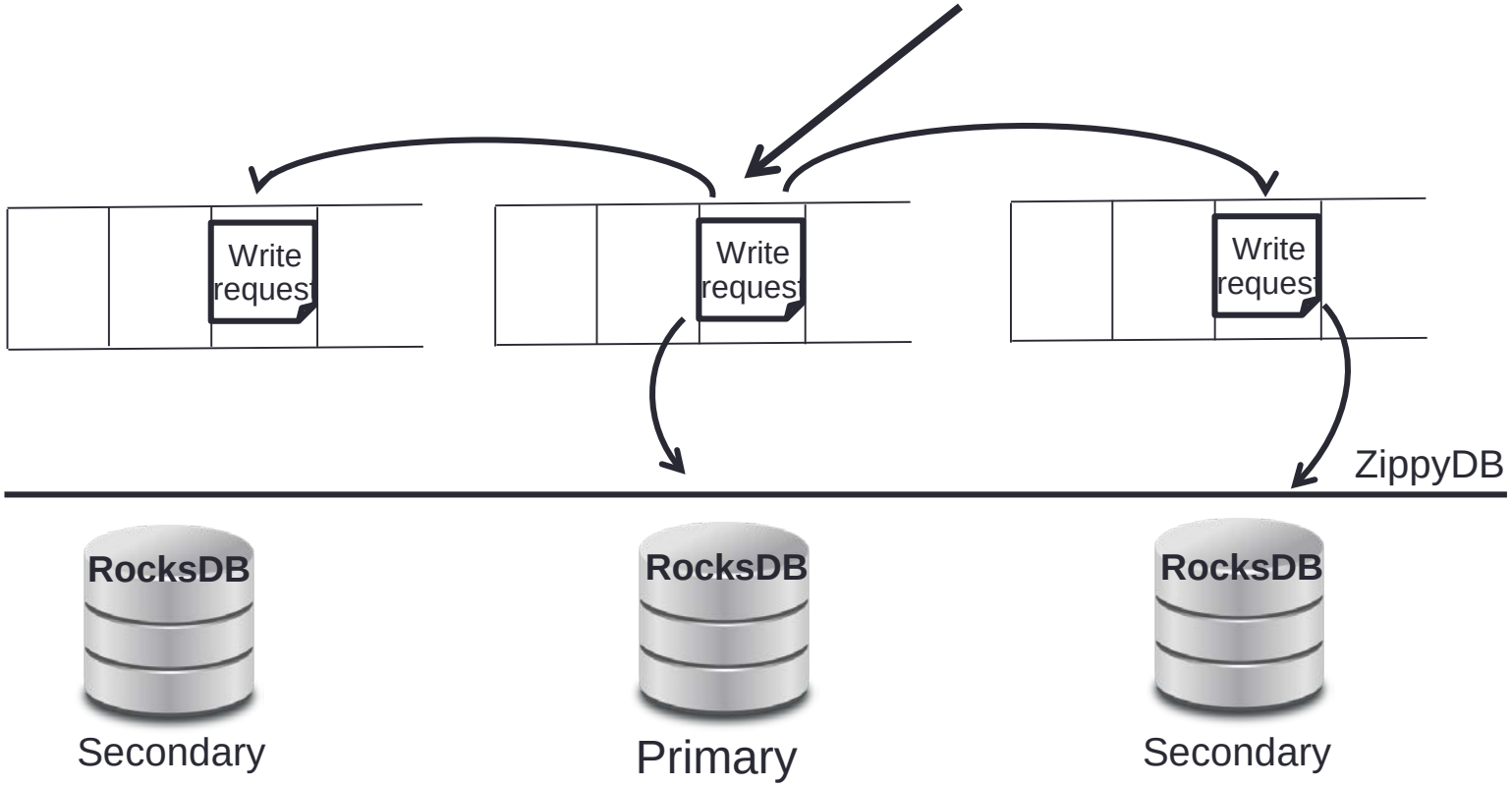
DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems
3. Safe recovery
   - ➔ With respect to system's consistency guarantees

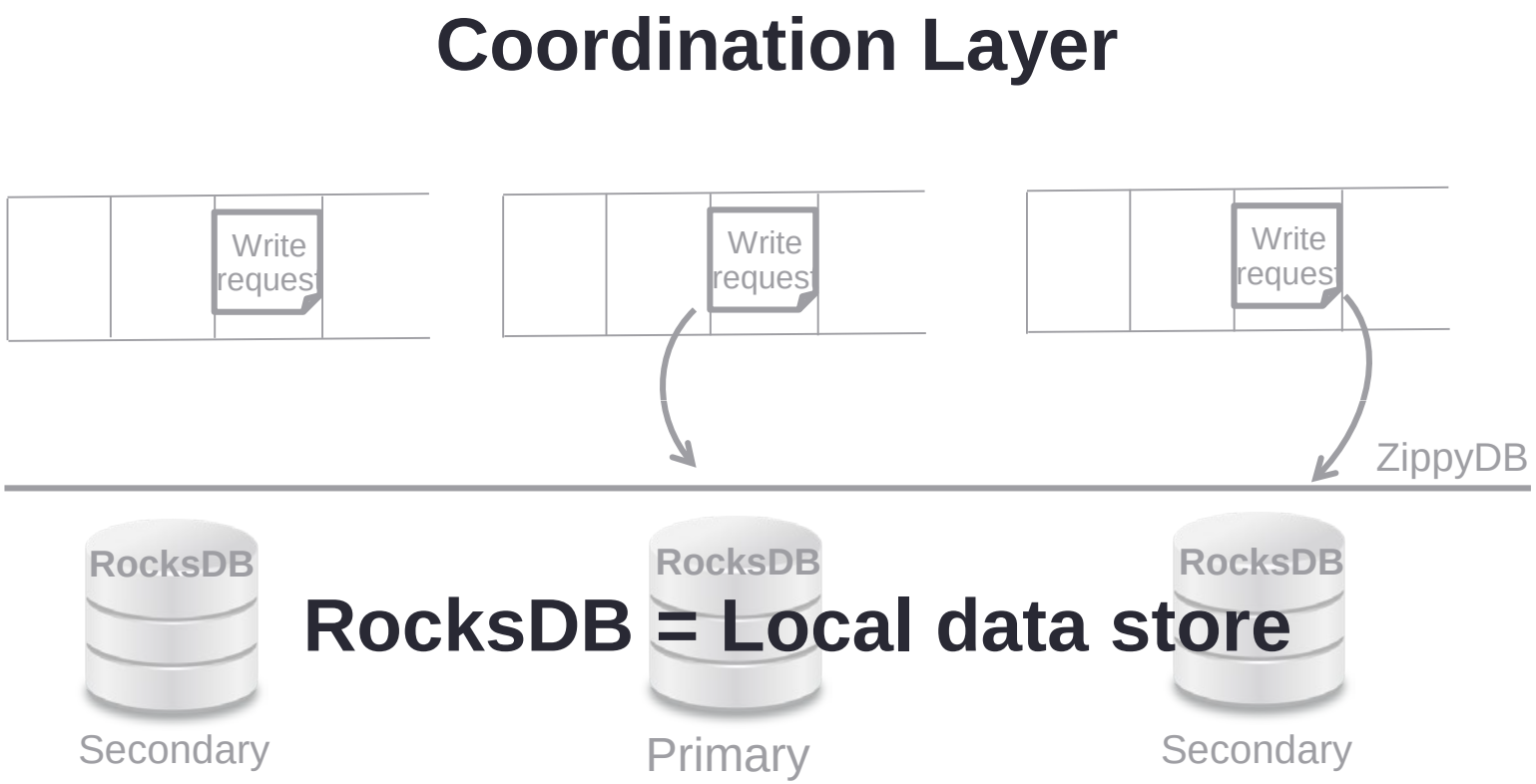
Implementations of DIRECT

- ZippyDB/RocksDB
  - RocksDB: KV store backed by log-structured merge tree
  - ZippyDB: distributed KV store backed by RocksDB
- HDFS: Block-level distributed file system

ZippyDB Overview



ZippyDB Overview



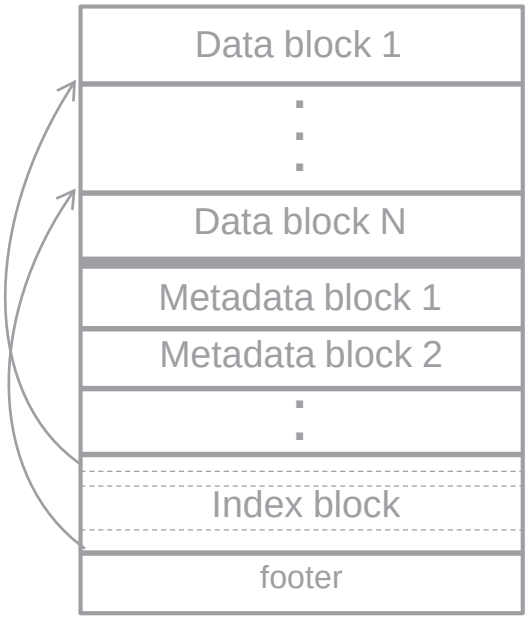
How ZippyDB handles corruptions

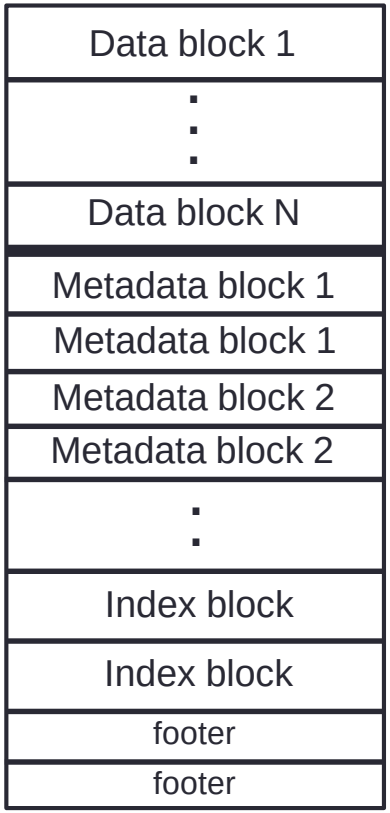
- User reads: retry from another server
- **Background reads (compaction): crash server**

ZippyDB-DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems
3. Safe recovery
   - ➔ With respect to system's consistency guarantees

RocksDB SST file layout



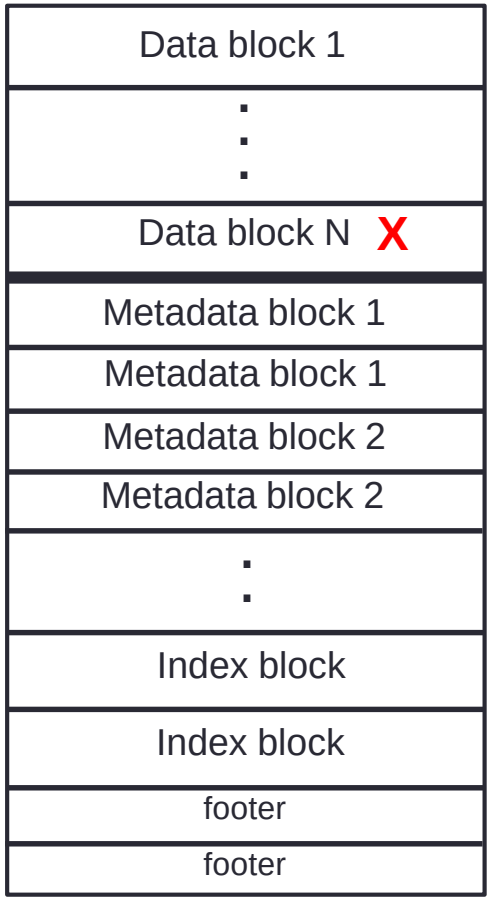


ZippyDB-DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems
3. Safe recovery
   - ➔ With respect to system's consistency guarantees

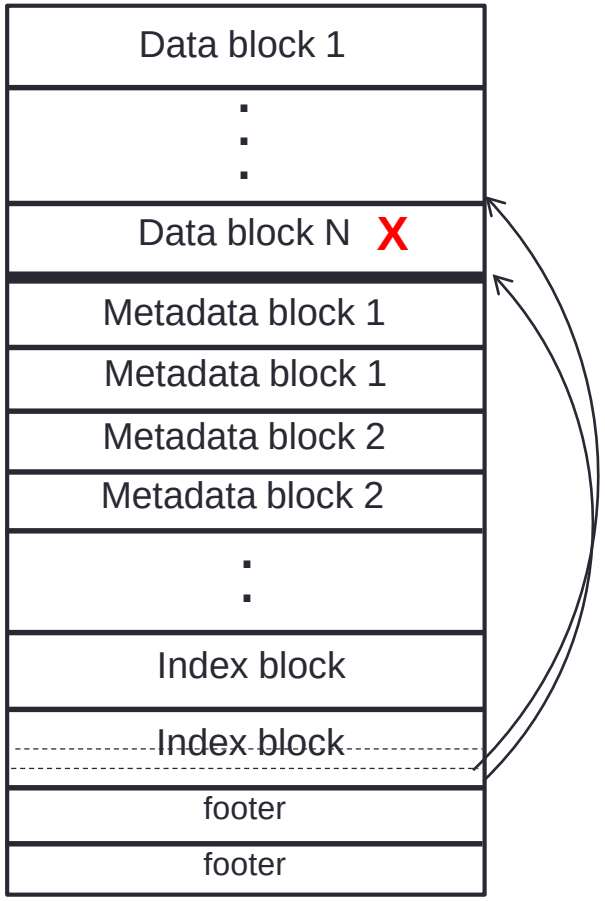
Identifying corrupt data

No way of knowing the exact key-value pair!



Index blocks identify corrupt range

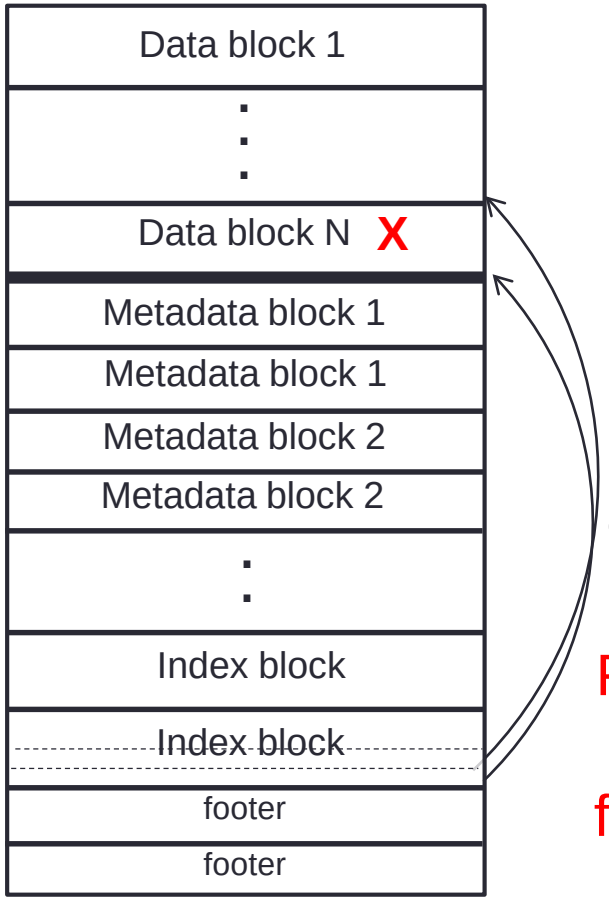
No way of knowing the exact key-value pair!



Instead.. use index entries to get a range

Index blocks identify corrupt range

No way of knowing the exact key-value pair!



<key123, value456>

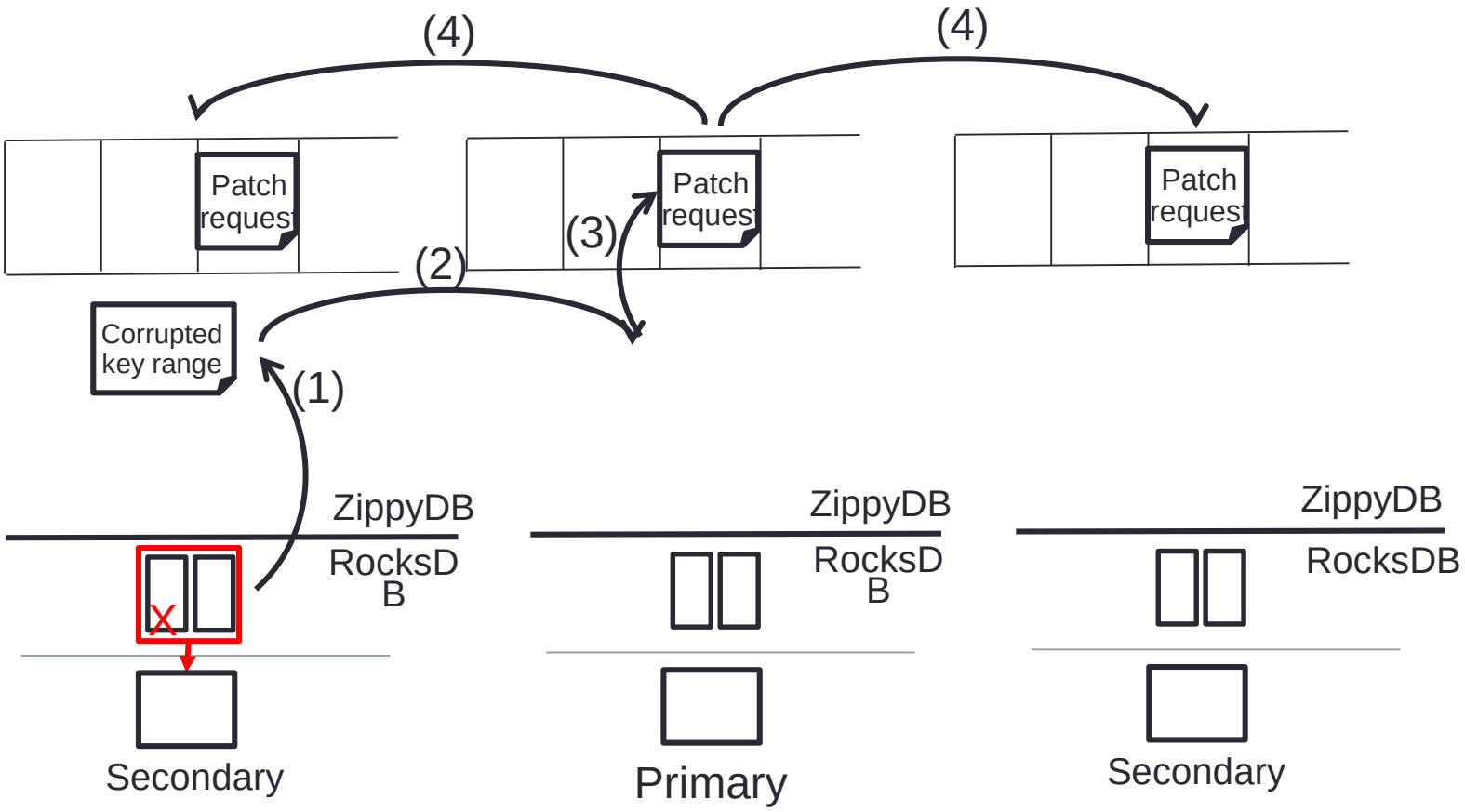
Instead.. use index entries to get a range

Fetch all keys in the range [key1,key2] from another replica

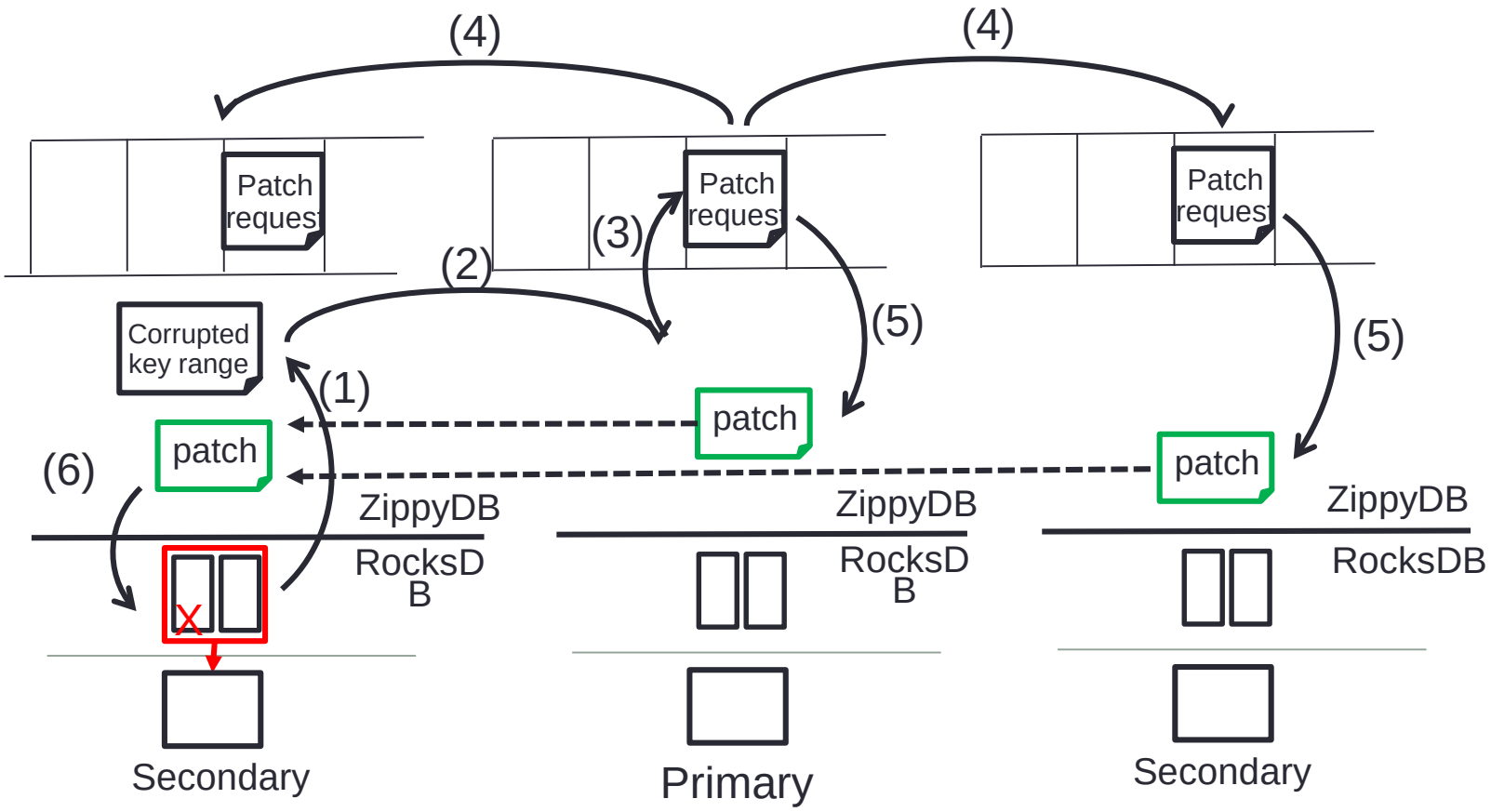
ZippyDB-DIRECT

1. Protect and fix errors in local metadata
   - ➔ With local replication of metadata
2. Fix errors in data objects with replicas
   - ➔ Minimize amount of data required from other replicas
   - ➔ Challenging in logically-replicated systems
3. Safe recovery
   - ➔ With respect to system's consistency guarantees

Safe recovery of corrupted data objects



Safe recovery of corrupted data objects



ZippyDB-DIRECT Summary

1. Replicate metadata blocks in-line
2. Recover data blocks by recovering key range
3. Run recovery through coordination layer for safety

ZippyDB-DIRECT Evalution

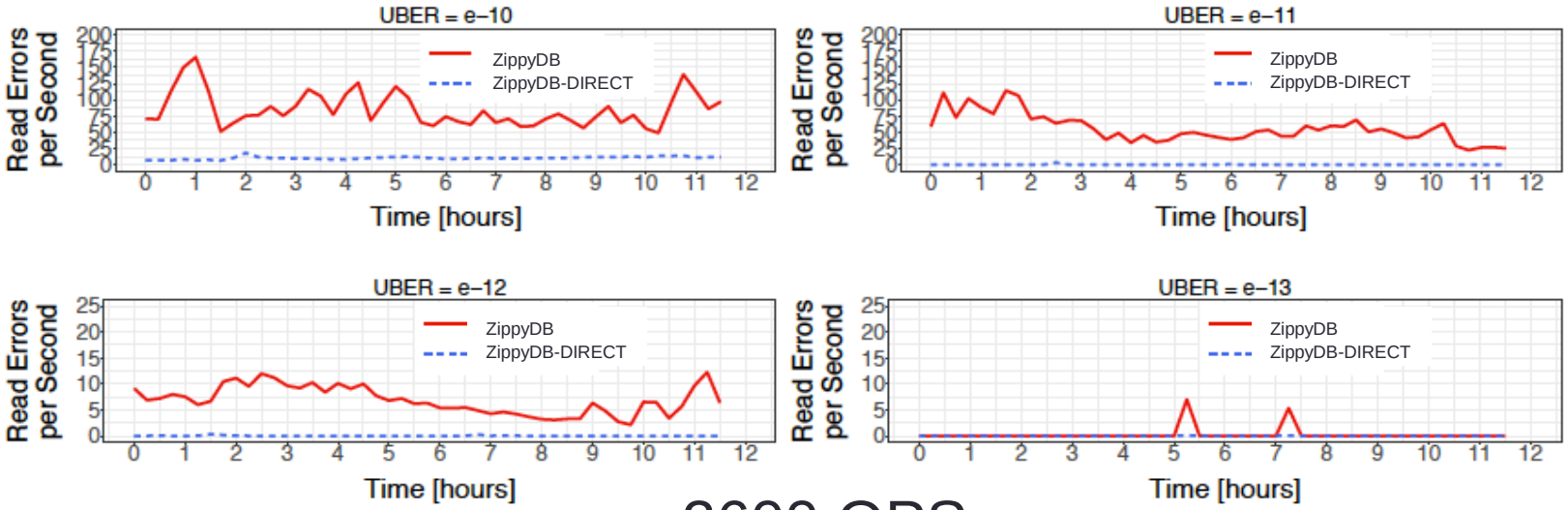
1. How much higher error rate can ZippyDB/RocksDB tolerate with DIRECT?

2. How much faster is recovery from bit corruptions in ZippyDB/RocksDB with DIRECT?

Experimental Setup

- 60 Facebook servers
- Traffic duplicated from live servers
- Error injection done on the fly in RocksDB

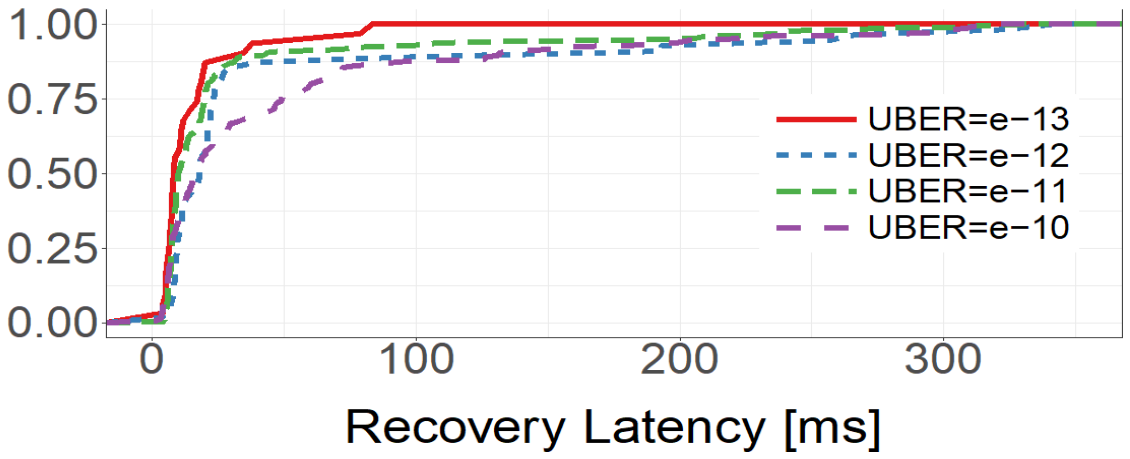
Error rate tolerance



~2600 QPS

UBER	10^{-10}	10^{-11}	10^{-12}	10^{-13}
ZippyDB	2.731%	1.981%	0.265%	0.011%
ZippyDB-DIRECT	0.187%	0.040%	0.0008%	0.0002%

DIRECT takes ms to recover, while ZippyDB takes **minutes**



Conclusion

- With *existing* distributed redundancy, DIRECT enables storage systems to:
  - Tolerate 100,000x higher bit error rates
  - Reduce recovery time from bit errors by orders of magnitude
- Two novel mechanisms:
  - Minimize the size of the recovered data
  - Leverage the coordination layer for safe recovery
- Consequently, extend flash lifetime by allowing devices to expose bit errors