KubeCon | CloudNativeCon

North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

KubeCon | CloudNativeCon
North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

How we make TiKV - a distributed storage, more cost-effective on the cloud

Andy (Yang) Zhang, Sunny Bains

KubeCon | CloudNativeCon
North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

How we make TiKV - a distributed storage, more cost-effective on the cloud

KubeCon | CloudNativeCon

North America 2022

BUILDING FOR THE ROAD AHEAD

DETROIT 2022

October 24-28, 2021

Andy (Yang) Zhang
Senior Software Engineer
PingCAP

Sunny Bains
Senior Director
PingCAP

Agenda

- Introduction to TiKV
- The cost of building a SaaS
- Reducing the costs
    - Reducing the computational cost
    - Reducing the storage cost
    - Reducing the network cost

What’s TiKV

A graduated project of the Cloud Native Computing Foundation (CNCF).

An open-source, distributed, and transactional key-value database.

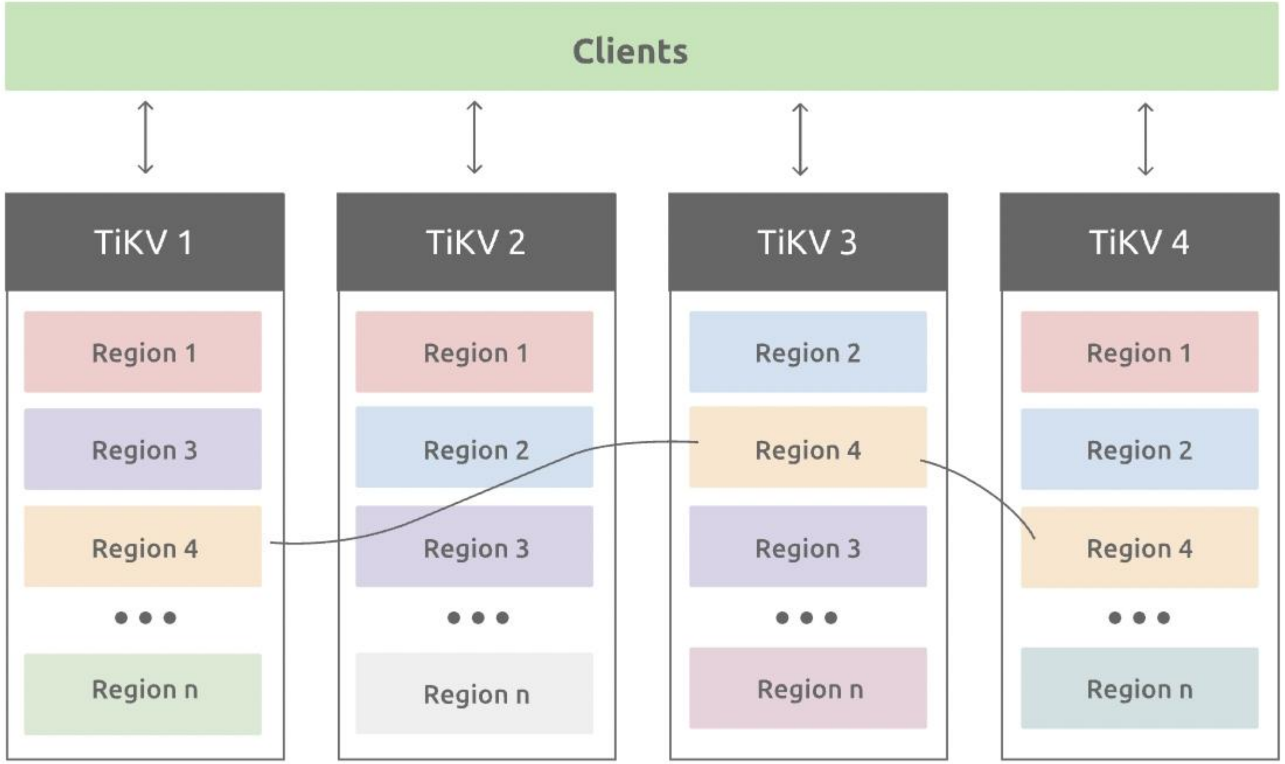
CLOUD NATIVE COMPUTING FOUNDATION

TiKV

github.com/tikv/tikv

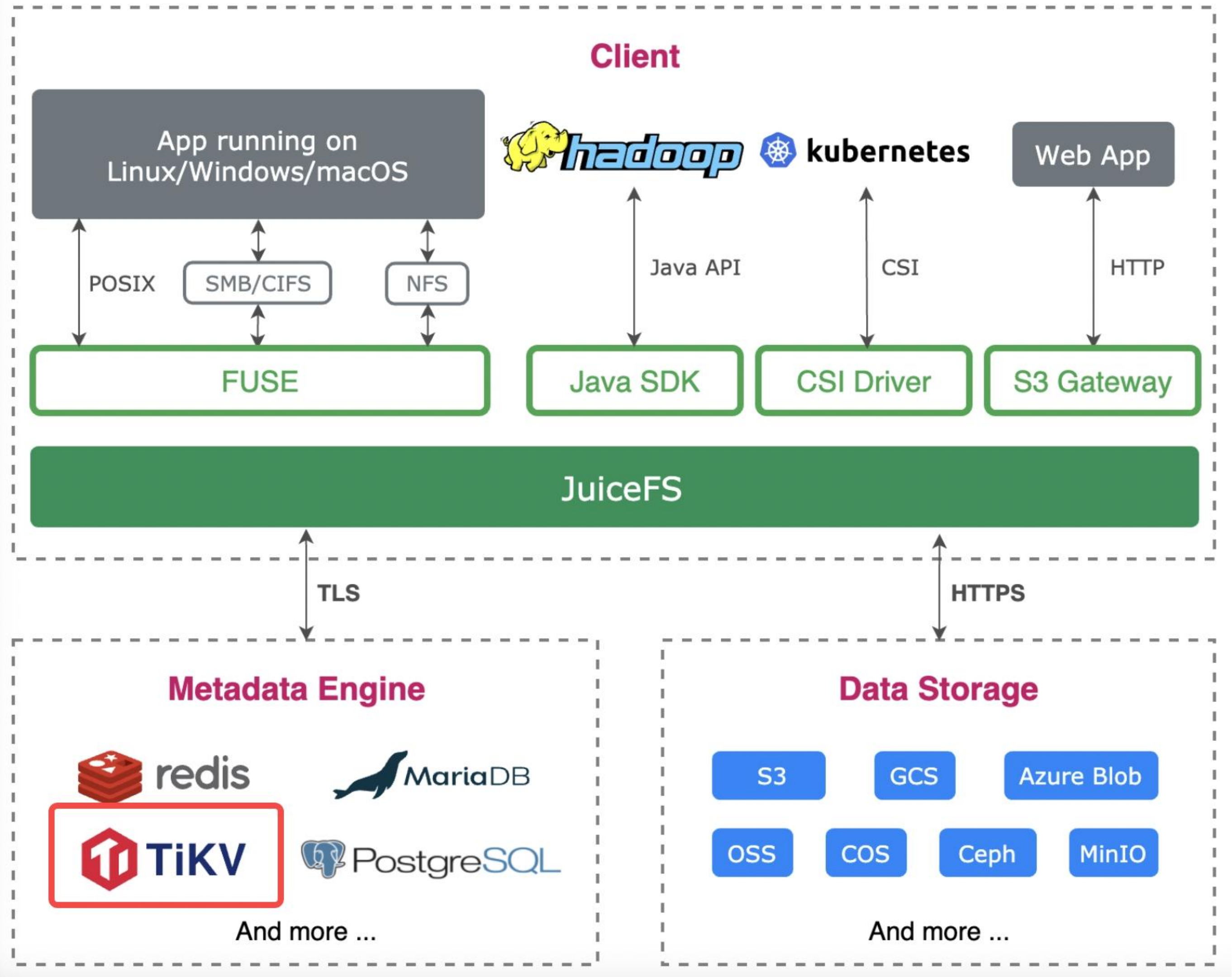
TiKV Architecture

- High Availability: Raft
- Horizontal Scalability: Automatic data range splitting|merging & balancing
- RawKV API & Transactional KV API(ACID)

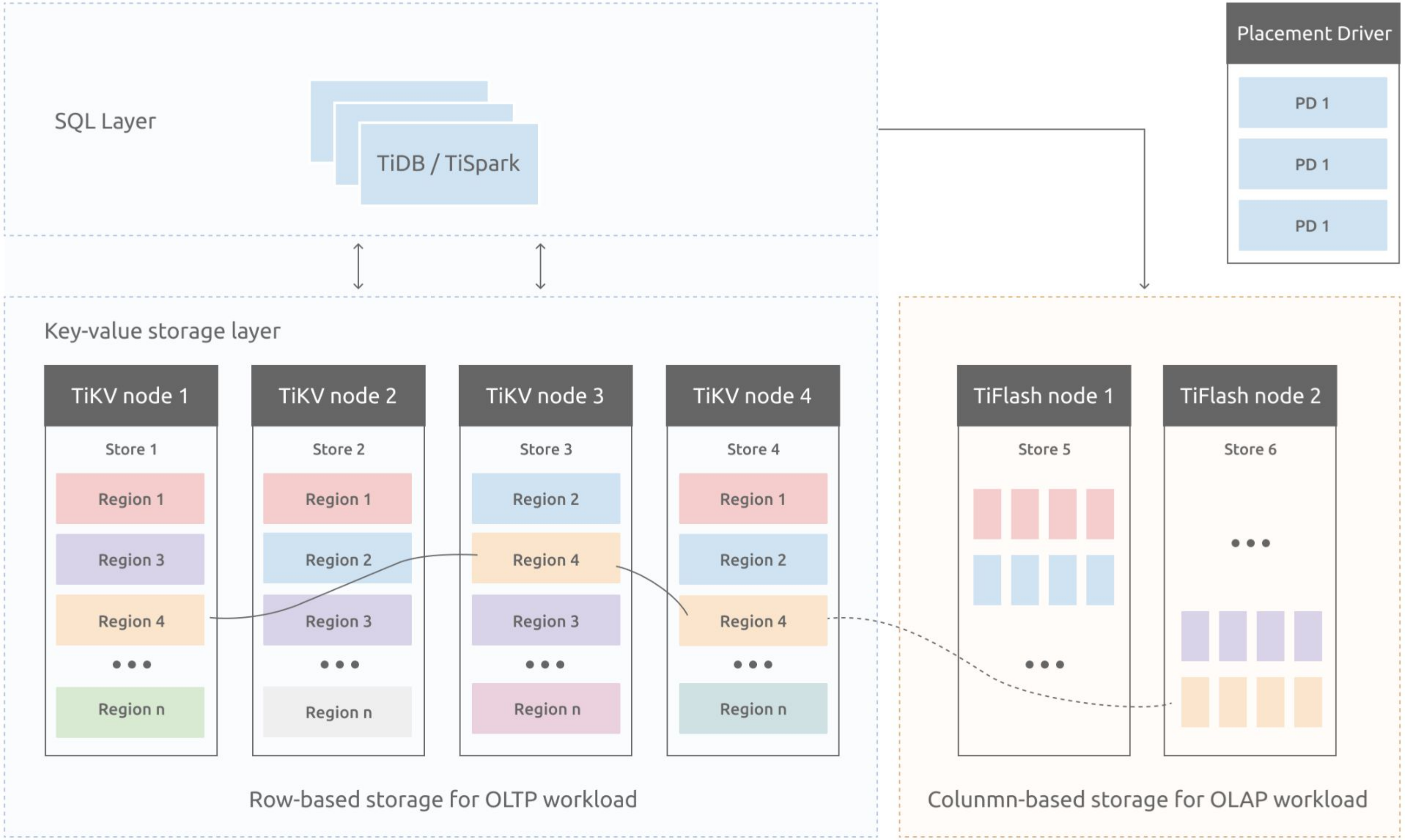


TiKV is being used widely

- TiKV as metadata storage: **JuiceFS** metadata, **JD Cloud** Object Storage metadata
- TiKV stores block-chain data: **Harmony**
- Transactional KV: **Niantic (Pokemon Go)**
- Redis protocol on top of TiKV: **tidis**, **titan**
- Use TiKV as database storage layer: **TiDB**



TiDB Architecture



Move to cloud

On-premises vs Cloud from a SaaS provider perspective

On-premises

pros:

- “free” hardwares
- complete control

cons:

- hard to support

Cloud

pros:

- anywhere, anytime
- scalable support services

cons:

- cost management

Nothing is free on cloud

- Computing resources (EC2, VM)
- Storage
  - Elastic Block Storage
    - Provisioned IOPS
    - Provisioned bandwidth
    - Storage
  - Object Storage
    - Storage
    - Requests
    - Data transfer
- Network
  - Data transfer (cross AZ, region)
  - NAT Gateway

TiDB Cloud cost analysis

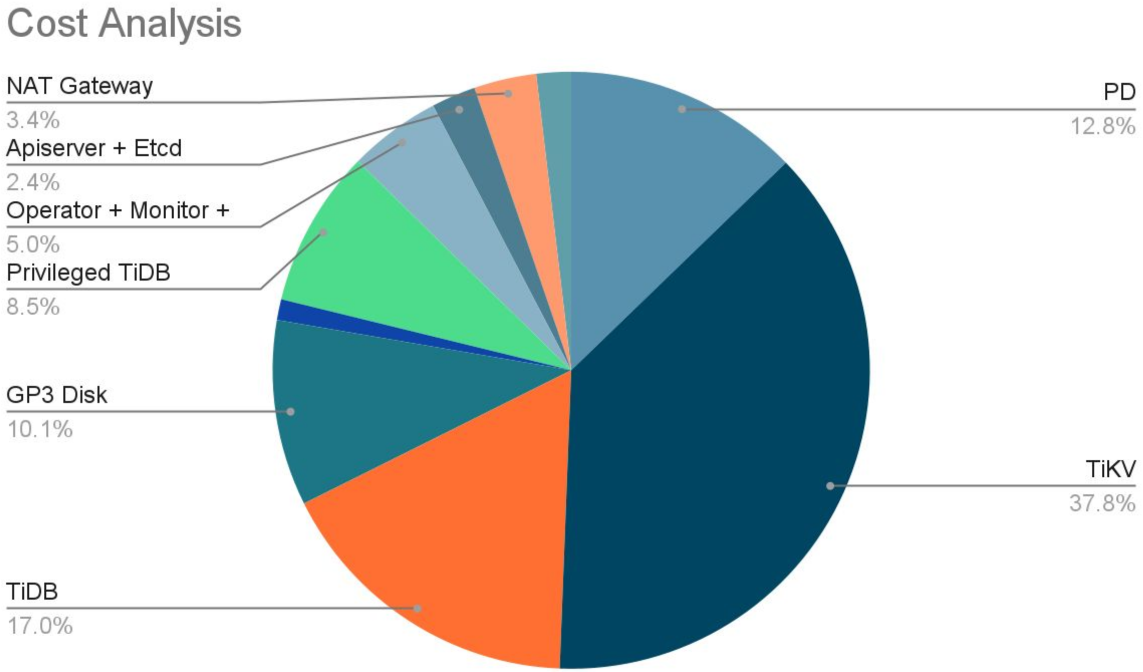
We have been doing cost analysis as much as we can

- by components
- by cloud services
- by regions
- by cloud vendors

Reducing the cost, making sure the resources are efficiently used is one of our top priorities.

Why?

Less cost, higher profit margin…

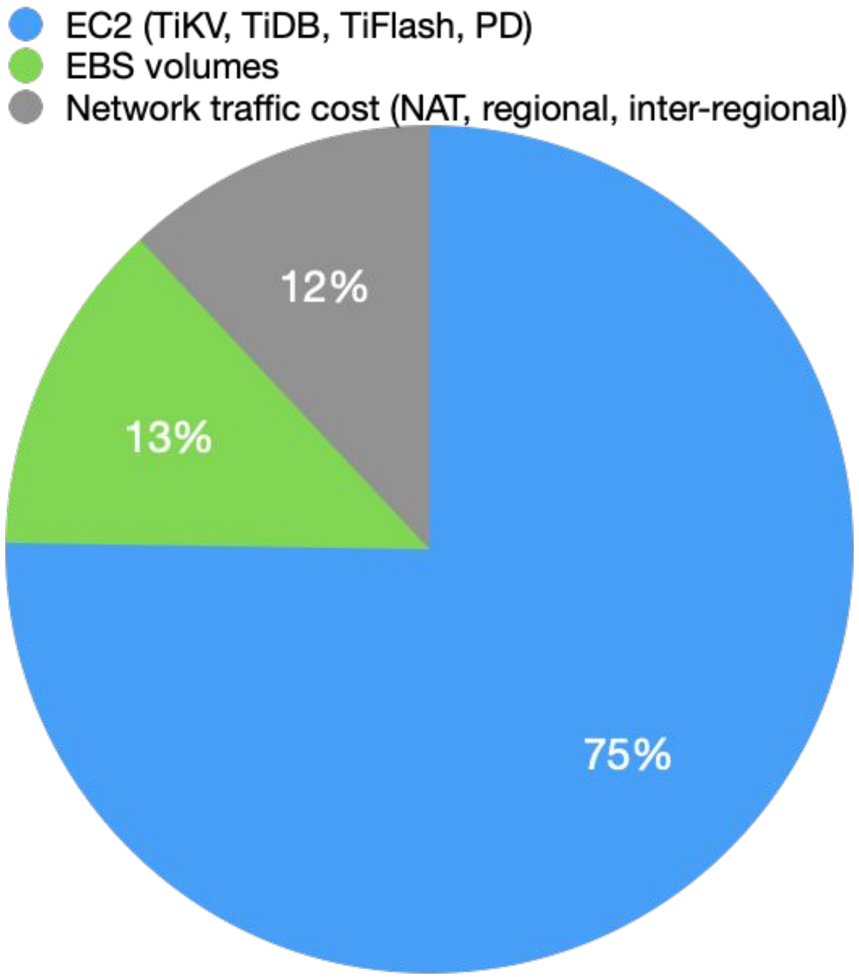


What can we do to reduce the cost

- Business level
  - Saving plan
- Technical level
  - Operational
    - Reclaim unused resources
  - Increase the utilization of provisioned resources
    - Vertical scale up / down
  - **Architecture, implementation**
    - Tools, e.g. monitoring, logging
    - **Main components**

Cost analysis - zoom in

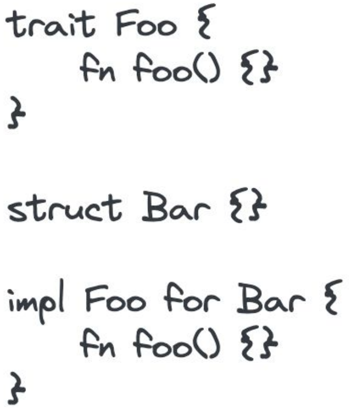
- EC2, VM
  - more efficient code
  - less unnecessary processes
- EBS, Persistent Disk
  - smaller size
  - less IO
  - smaller bandwidth
- Network
  - less cross-region traffic
  - less cross-available-zone traffic



Rust trait static dispatch

	pros	cons
static dispatch (monomophirization, generics)	better performance	type propagation
dynamic dispatch (polymorphism)	better readability	vtable lookups

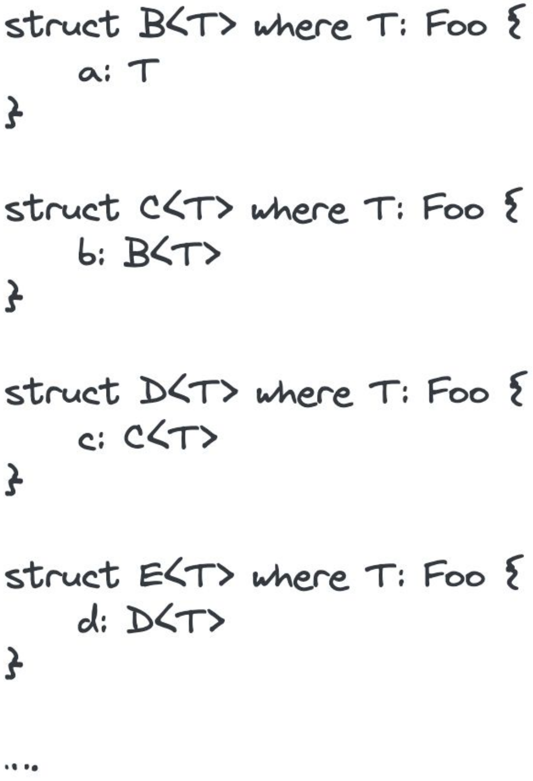
Type propagation:



```
trait Foo {
    fn foo() {}
}

struct Bar {}

impl Foo for Bar {
    fn foo() {}
}
```



```
struct B<T> where T: Foo {
    a: T
}

struct C<T> where T: Foo {
    b: B<T>
}

struct D<T> where T: Foo {
    c: C<T>
}

struct E<T> where T: Foo {
    d: D<T>
}

....
```

Put inactive groups into sleep

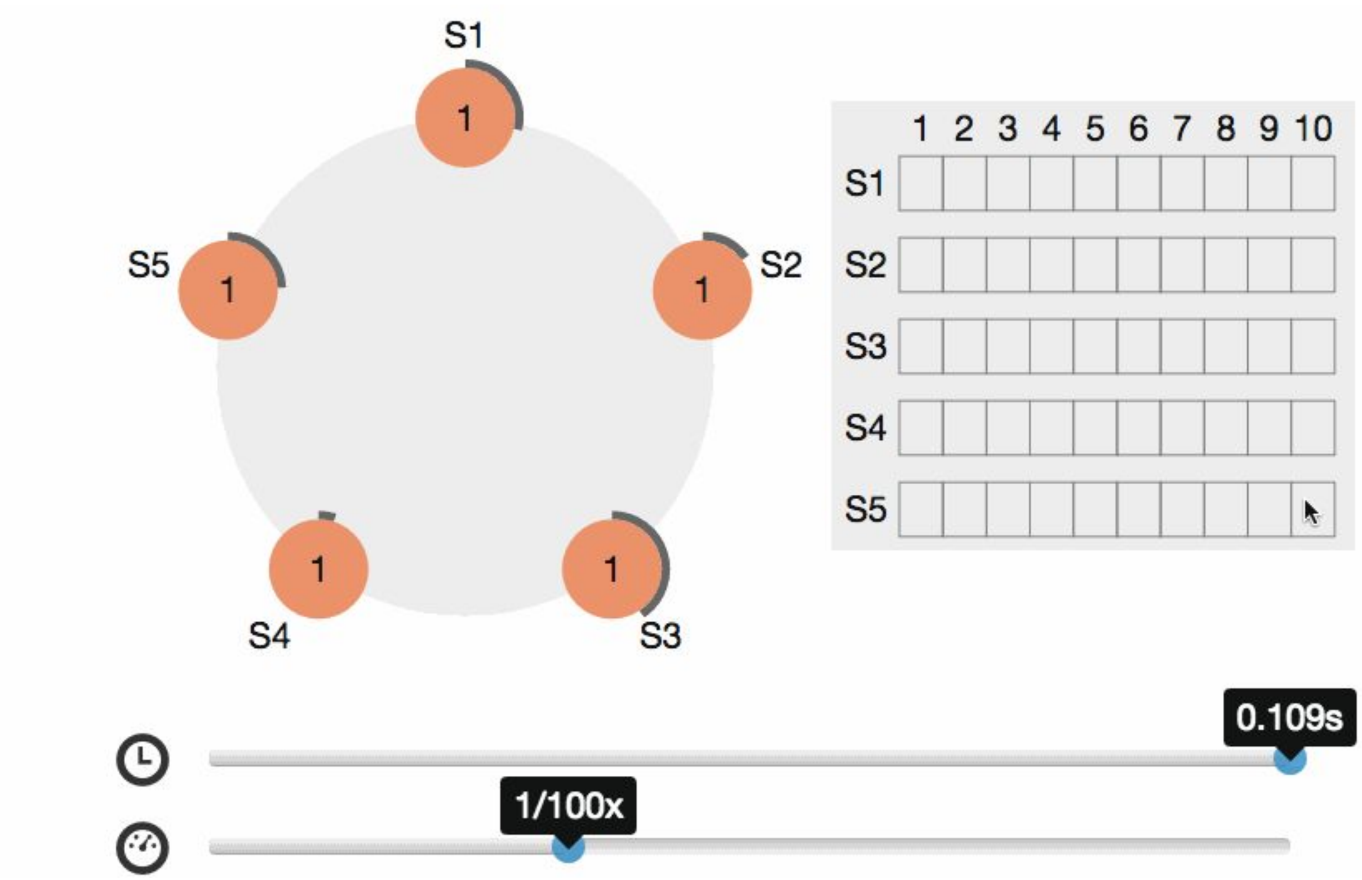
In Raft algorithm:

Leader periodically sends heartbeats to its follower (every 2 secs by default in TiKV).

This can be costly, especially when the cluster has a large number of Raft groups.

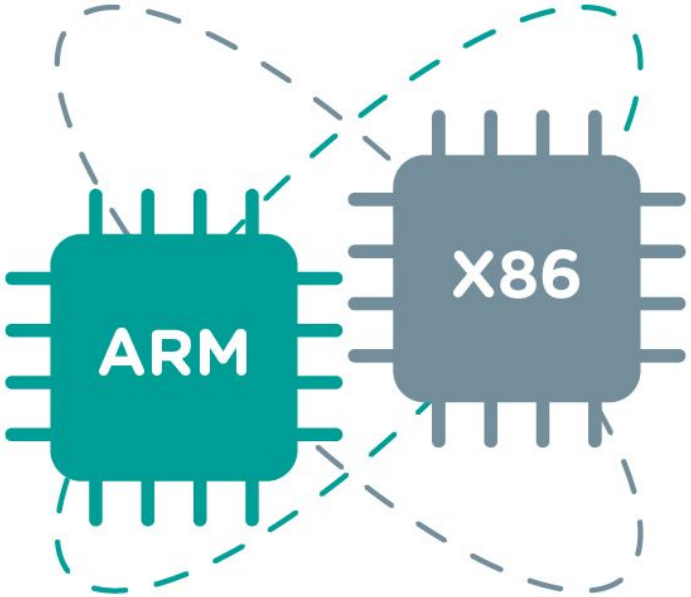
In reality, most groups are inactive.

Put them into sleep!



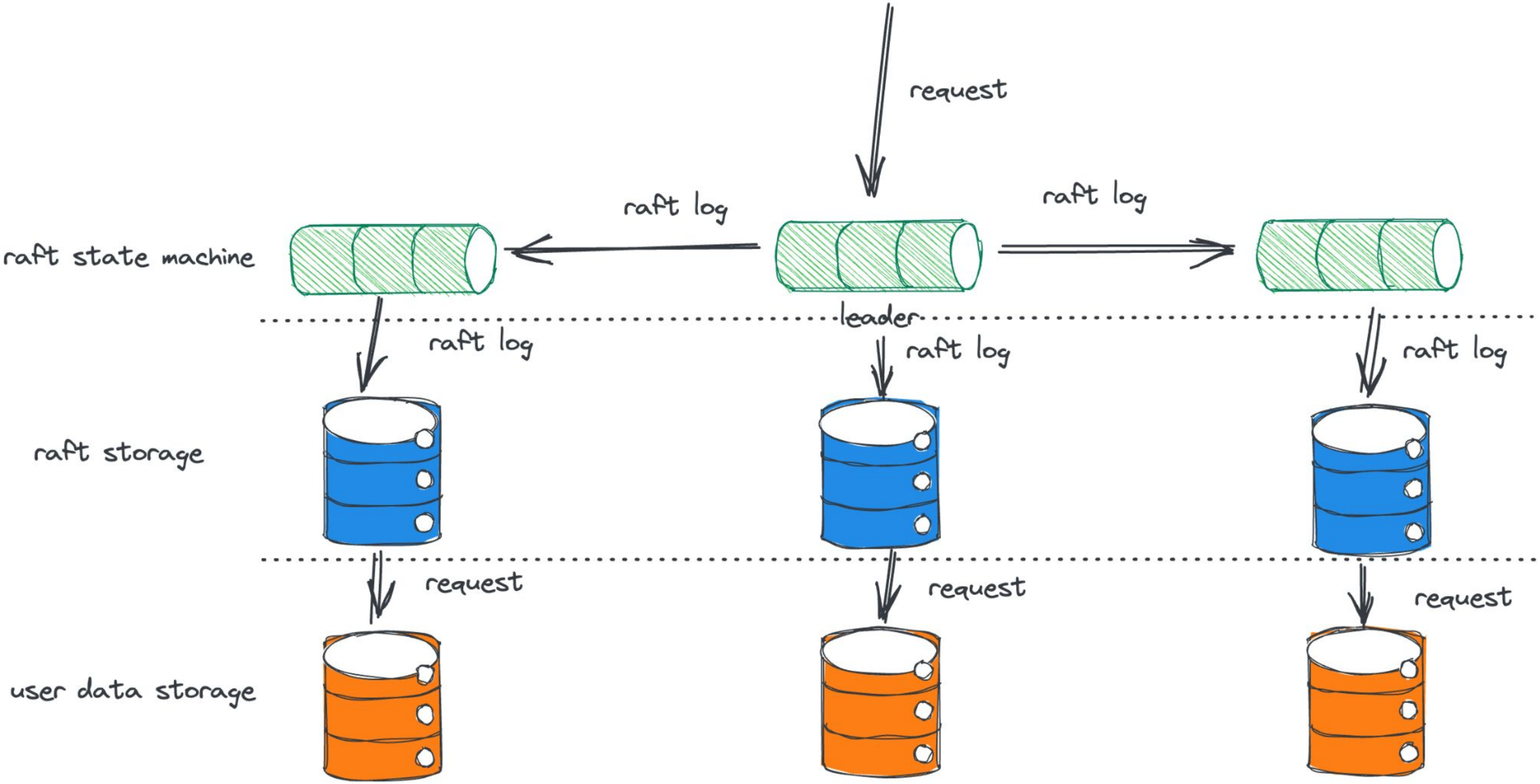
	1	2	3	4	5	6	7	8	9	10
S1										
S2										
S3										
S4										
S5										

ARM instead of AMD64 or X86-64

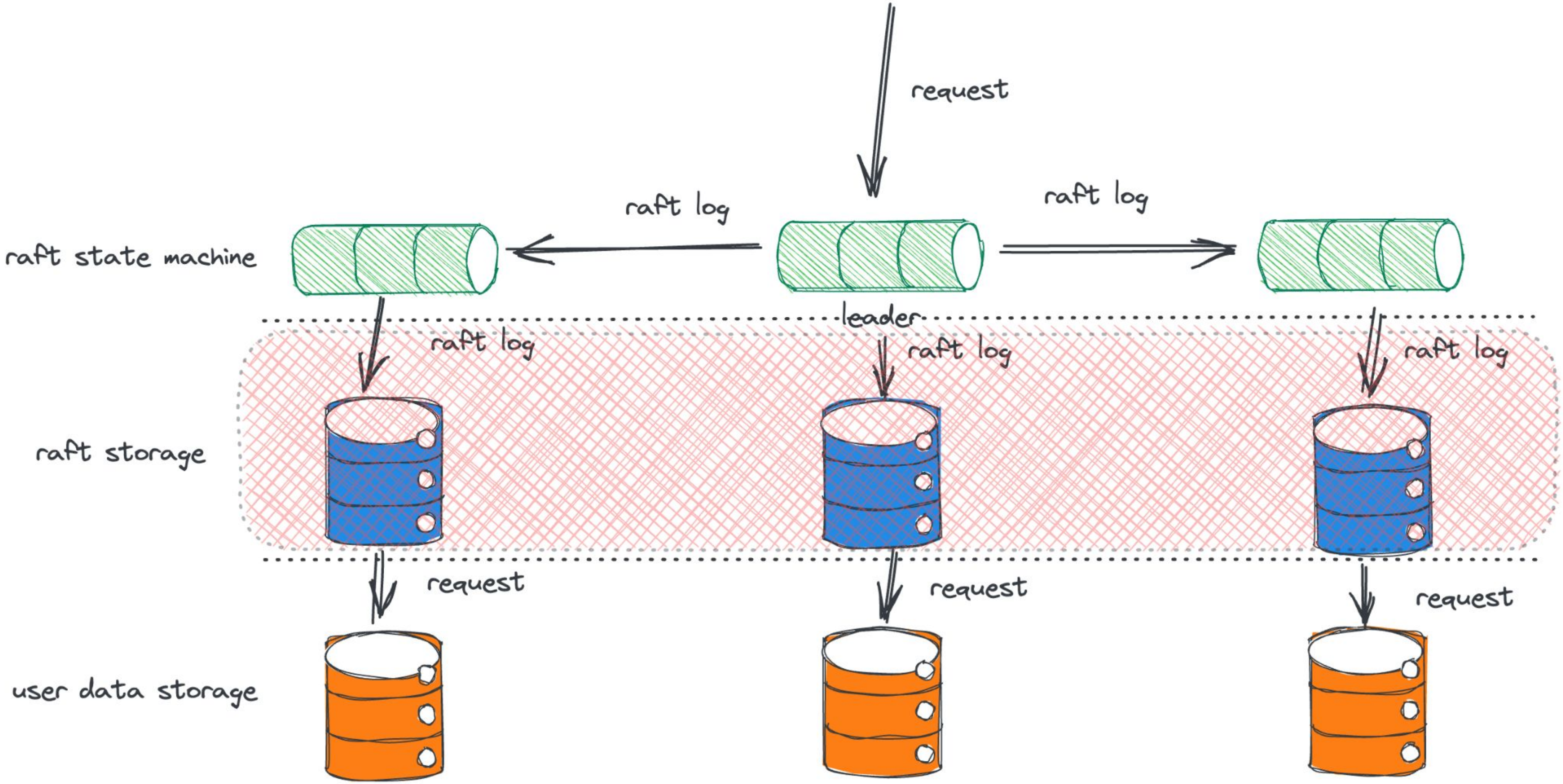


	x86	ARM
performance	100%	100%
cost	100%	80%

TiKV, under the hood



TiKV, under the hood



Raft Engine

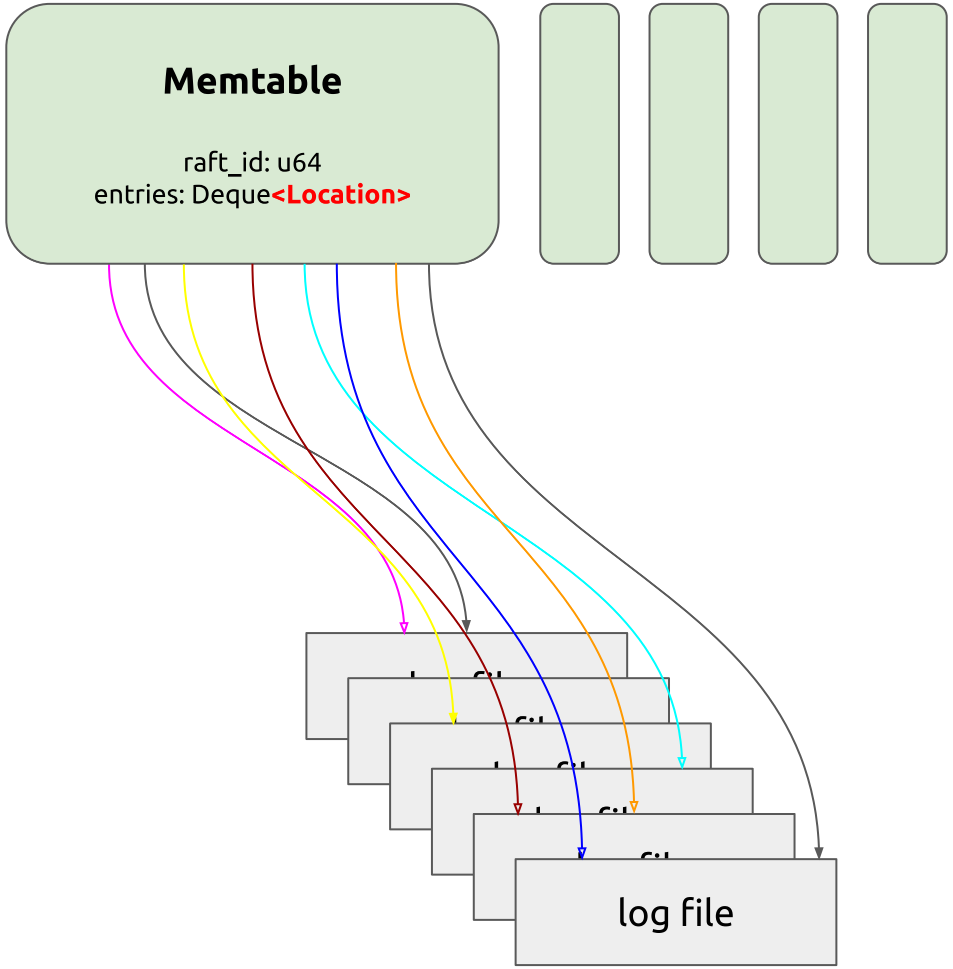
We created a dedicated storage engine to store Raft logs …

https://github.com/tikv/raft-engine

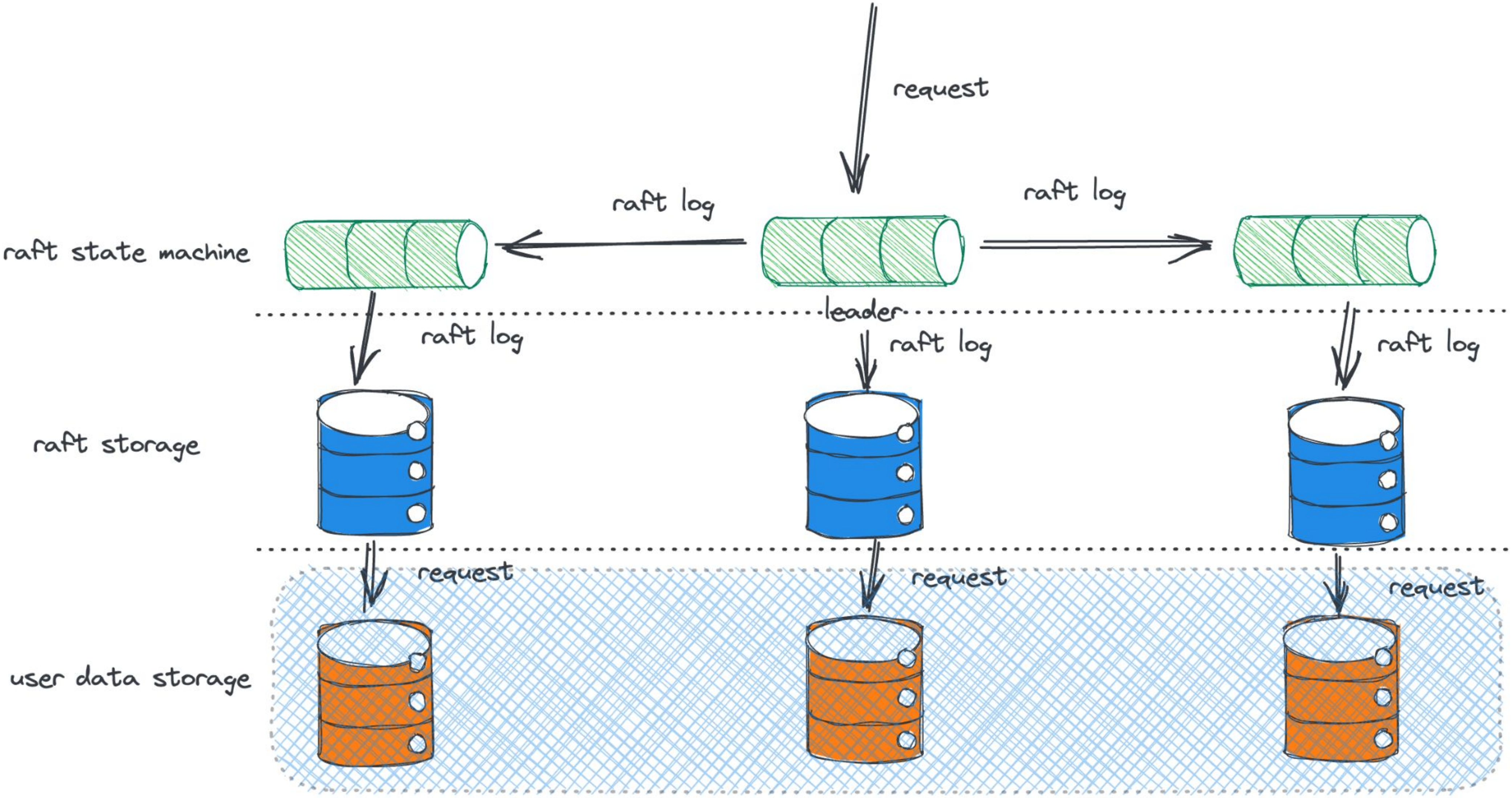
Why? Because Raft logs are guaranteed to be sequential

And, RocksDB is too “heavy” in this case…

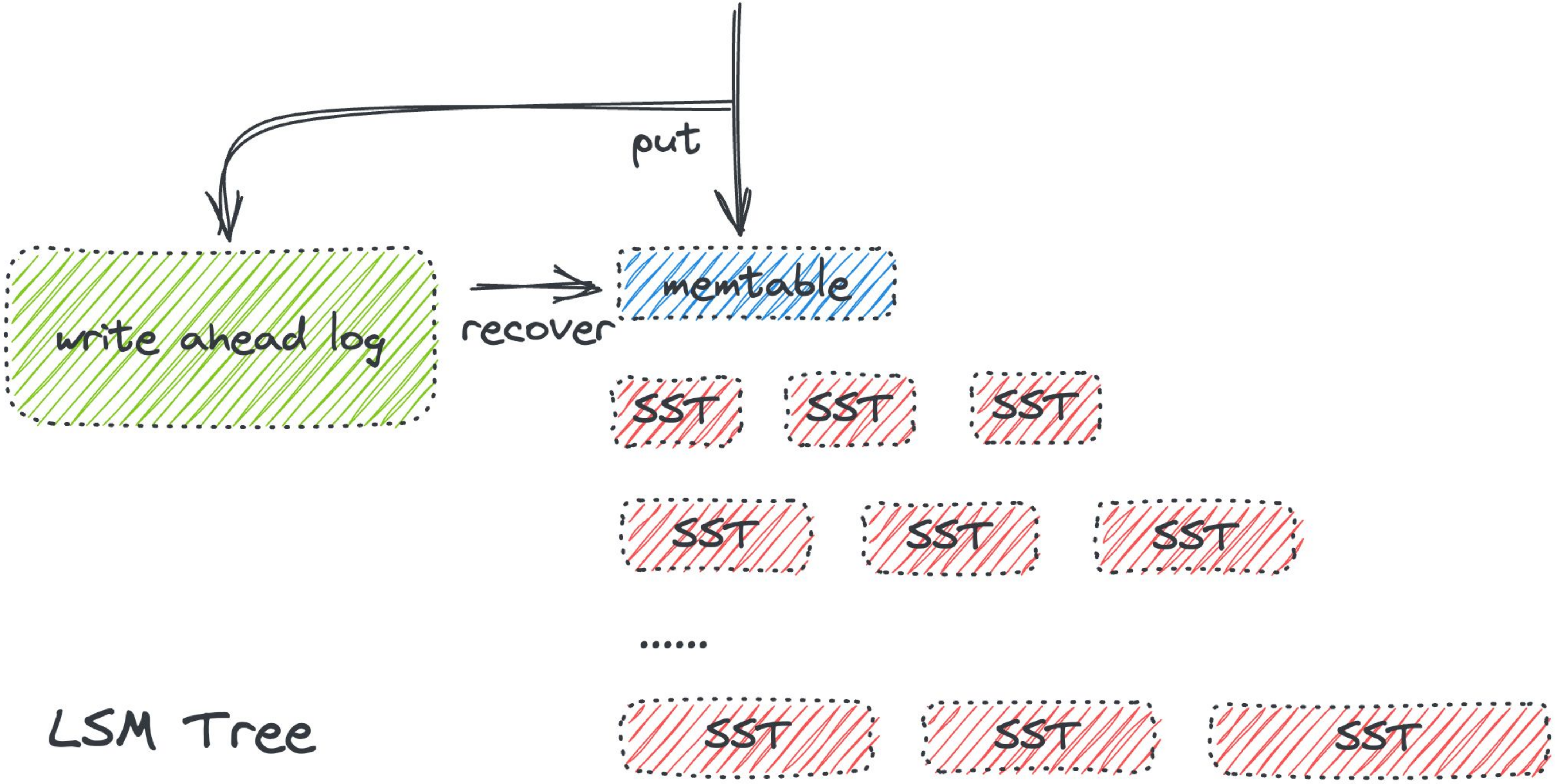
Through Raft Engine, we were able to greatly reduce the IOPS with a lower bandwidth. yay!



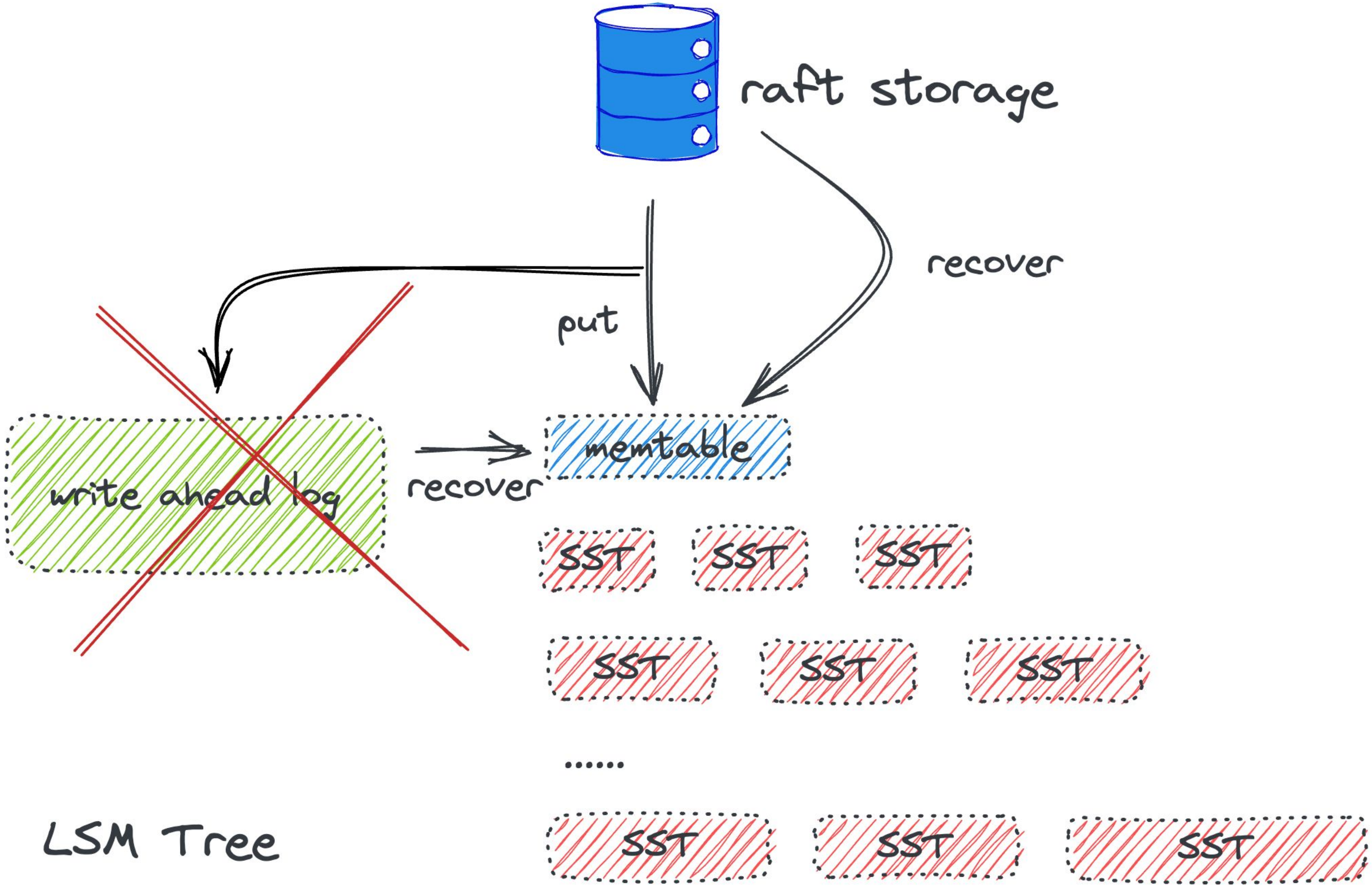
TiKV, under the hood



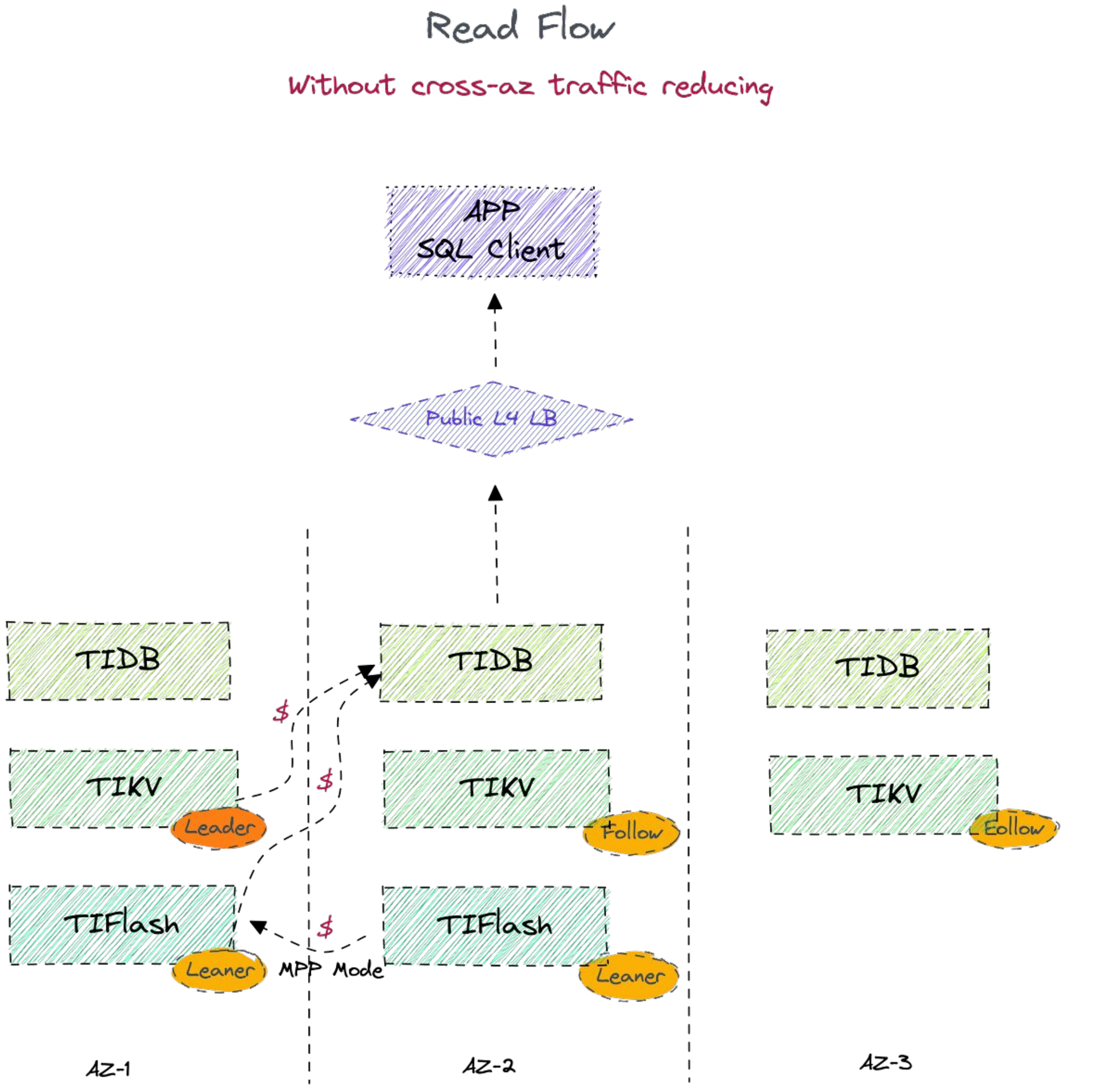
Disable RocksDB WAL

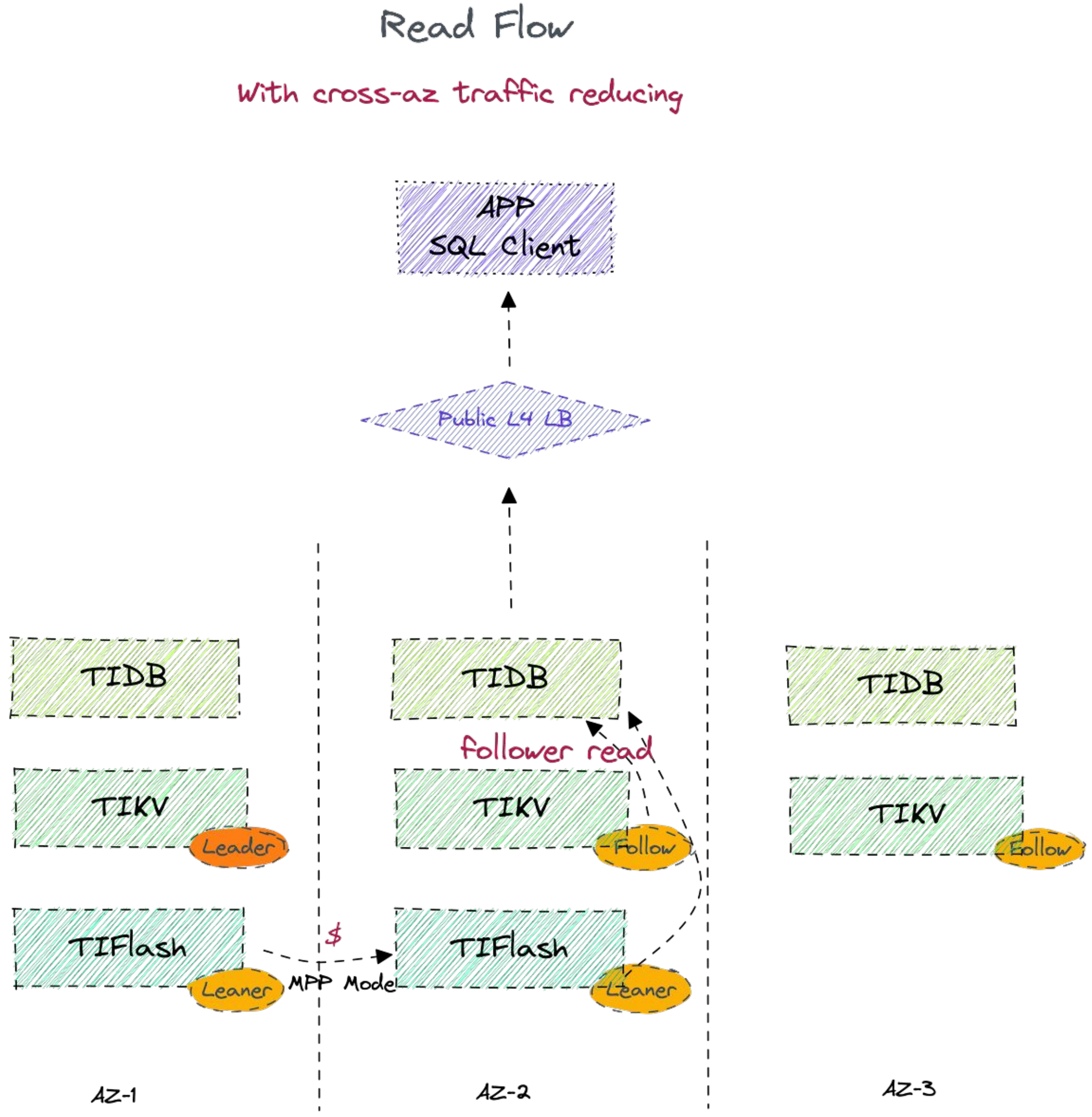


Disable RocksDB WAL



Reducing cross-AZ read traffic

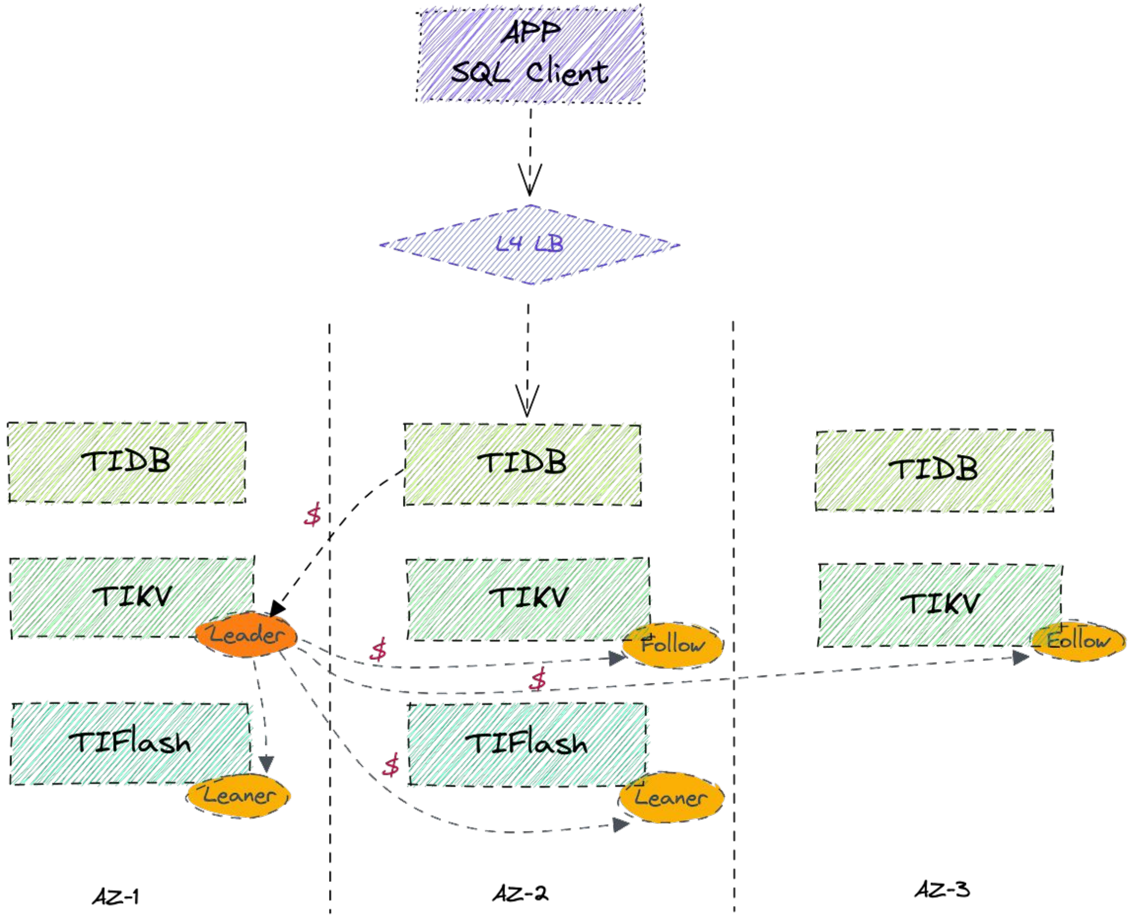




Reducing cross-AZ write traffic

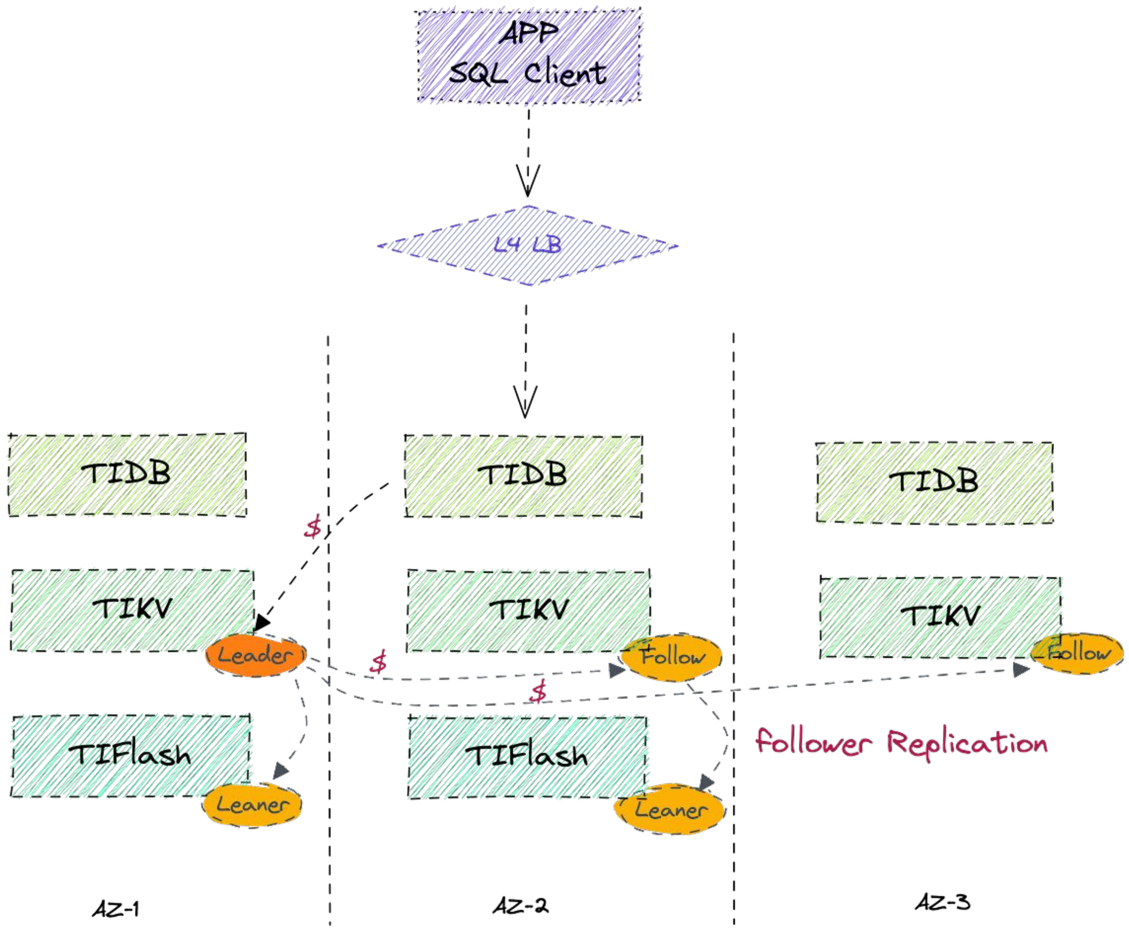
Write Flow

Without cross-az traffic reducing



Write Flow

With cross-az traffic reducing



Q&A

Please scan the QR Code above to leave feedback on this session