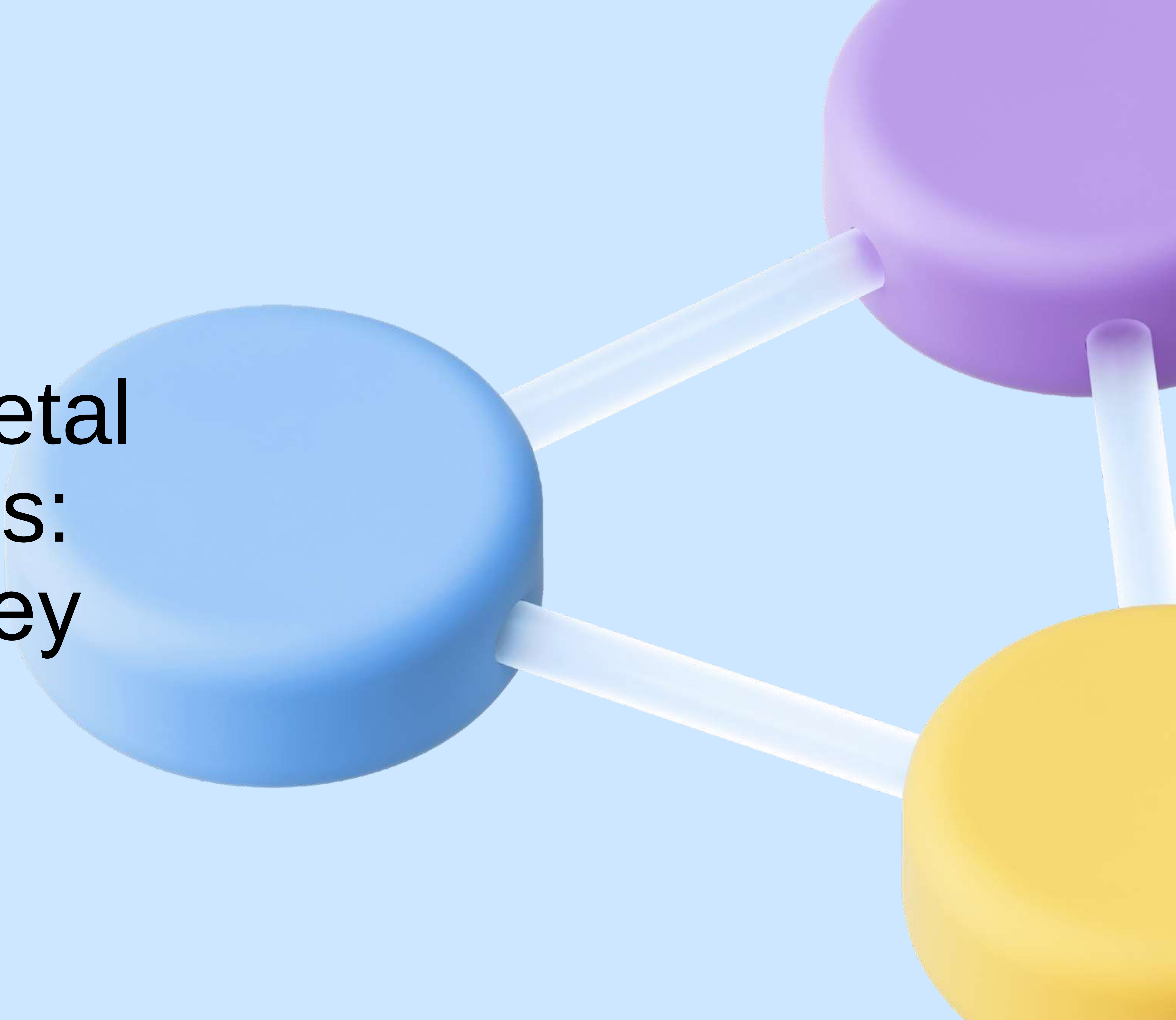
YDB

From Baremetal to Kubernetes: YDB’s Journey



Jorres Tarasov,
DevOps Engineer

NoSQL?

- Scalability
- High availability

NoSQL?

- Scalability
- High availability

- Lost atomicity
- Data anomalies
- Expressiveness

Maybe just run a relational database?

- This setup has a lot of community expertise and is battle tested
- You are happy as long as your data fits on a single node…

Maybe just run a relational database?

- This setup has a lot of community expertise and is battle tested
- You are happy as long as your data fits on a single node…
- … but then your developers turn into database administrators:

Let’s reshard our data tomorrow.

AGAIN??

What is YDB?

We have built an open-source distributed relational database that combines the **fault tolerance** and **scalability** of NoSQL **with strong consistency** and **feature-completeness** of traditional relational SQL databases

1. **ACID properties**
2. High availability, selfheal
3. Disaggregated storage\compute
4. Multitenancy

Expressive power of ACID

- Unconditional requirement
- Easier to think in terms of ACID
- Managed to have **distributed** ACID transactions

ACID stands for

- Atomicity
- Consistency
- Isolation
- Durability

1. ACID properties
2. **High availability, selfheal**
3. Disaggregated storage\compute
4. Multitenancy

Hardware failures

Our largest cluster

9700

nodes

32

petabytes

Hardware failures

Our largest cluster

9700
nodes

32
petabytes

Consider 1000 drives, MTTF* = 160 years →
MTTF of the first drive would be 8 days :(

* MTTF — mean time to failure

1. ACID properties
2. **High availability, selfheal**
3. Disaggregated storage\compute
4. Multitenancy

Hardware failures

Our largest cluster

9700 nodes

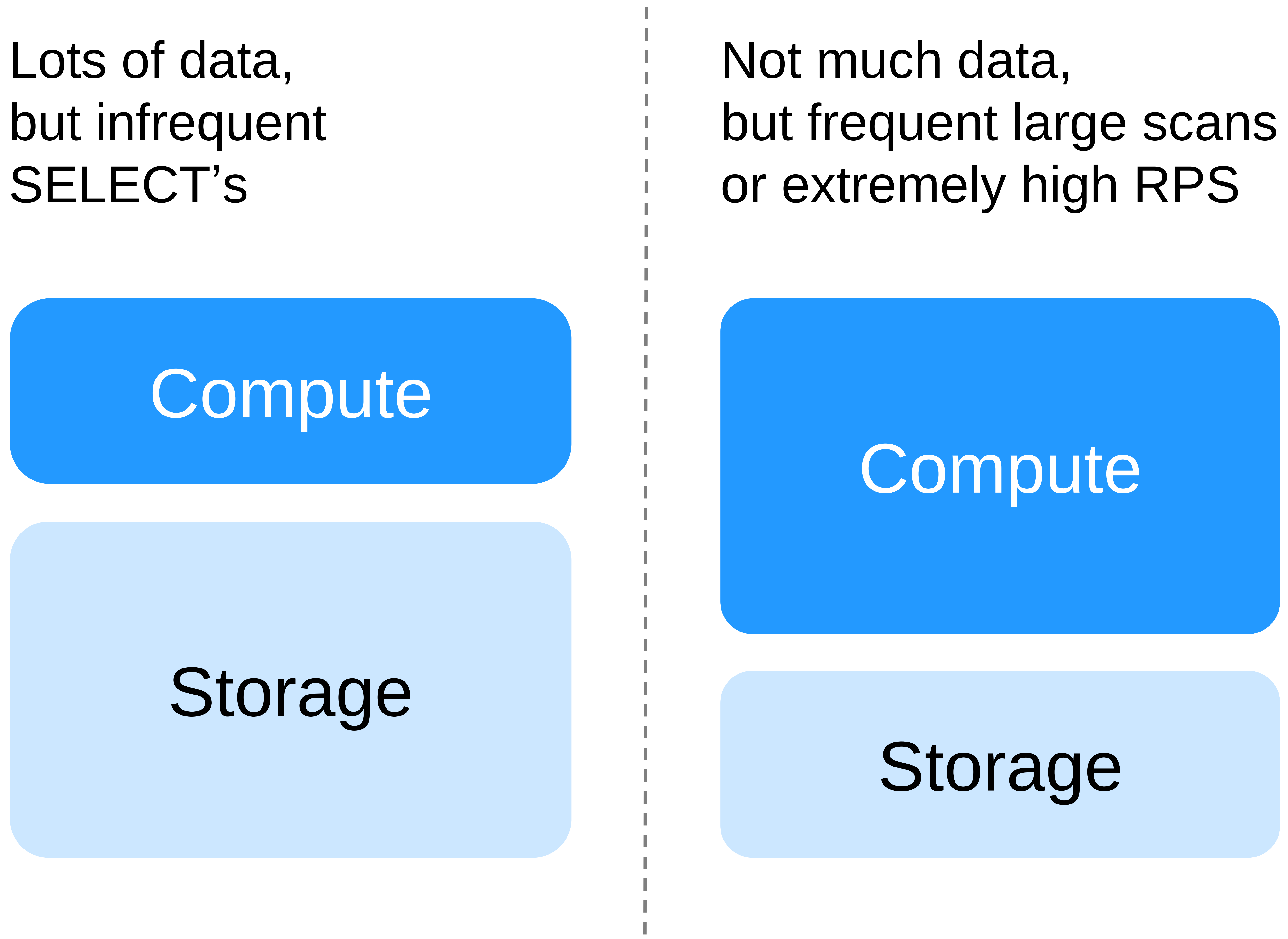
32 petabytes

With self-heal, we can unplug the drive anytime

The data would automatically re-persist to vacant nodes to maintain availability guarantees

1. ACID properties
2. High availability, selfheal
3. **Disaggregated storage\compute**
4. Multitenancy

Catering to different needs



1. ACID properties
2. High availability, selfheal
3. Disaggregated storage\compute
4. **Multitenancy**

Even though it’s hard…

Access isolation and resource quotas

1. ACID properties
2. High availability, selfheal
3. Disaggregated storage\compute
4. **Multitenancy**

Even though it’s hard…

Access isolation and resource quotas

Non-multitenant world: what does it take to run N Postgres’es?

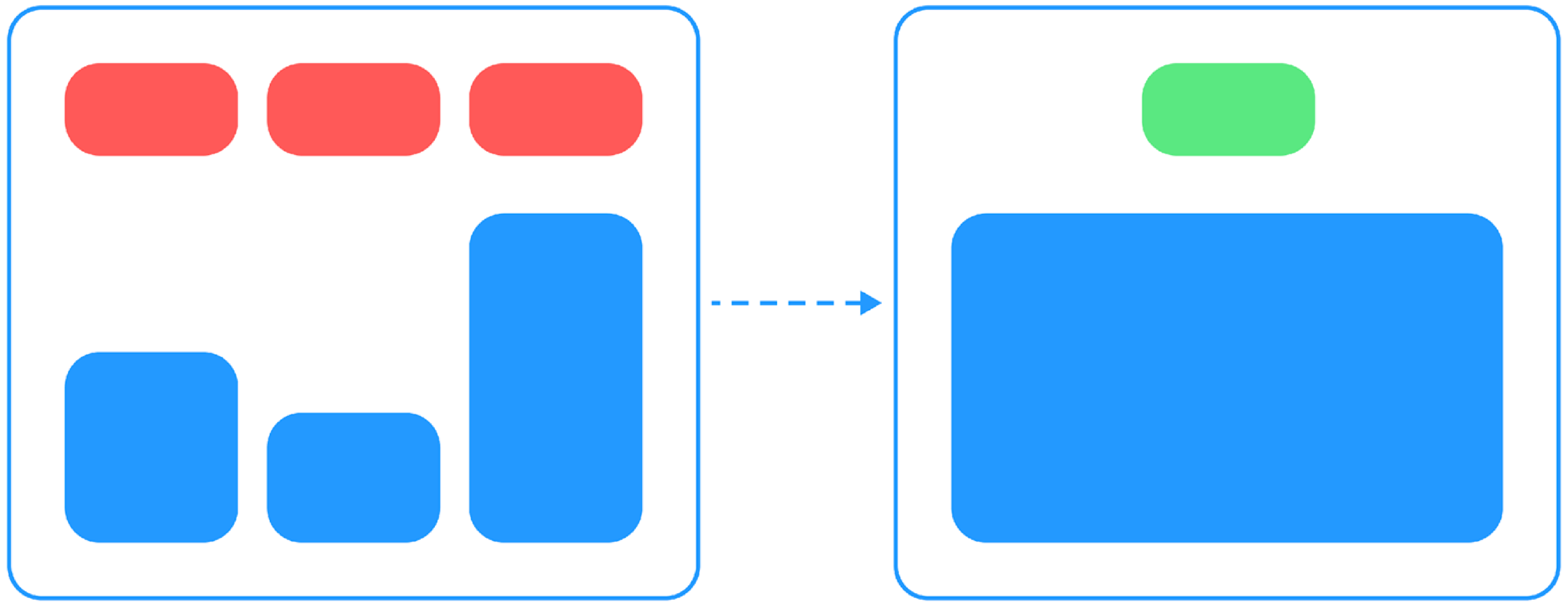
1. ACID properties
2. High availability, selfheal
3. Disaggregated storage\compute
4. **Multitenancy**

Even though it’s hard…

Access isolation and resource quotas

Non-multitenant world: what does it take to run N Postgres’es?

Suboptimal resource utilization:



Part 2,
migrating to K8s

The pillars of Kubernetes

1

Autoscale

2

Autoheal

3

Ecosystem

Nigel Poulton approves…

The pillars of Kubernetes

1

Autoscale

2

Autoheal

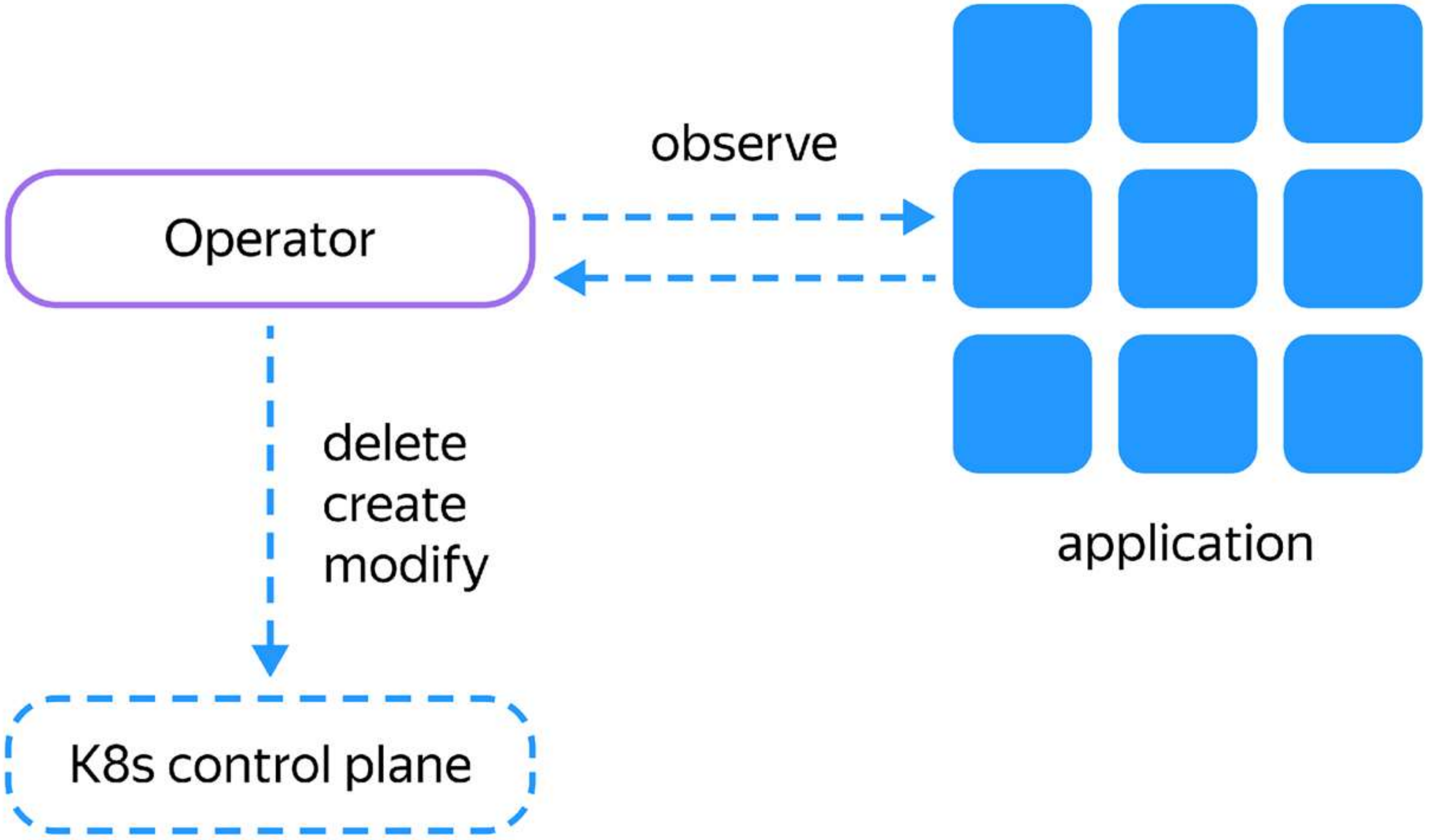
3

Ecosystem

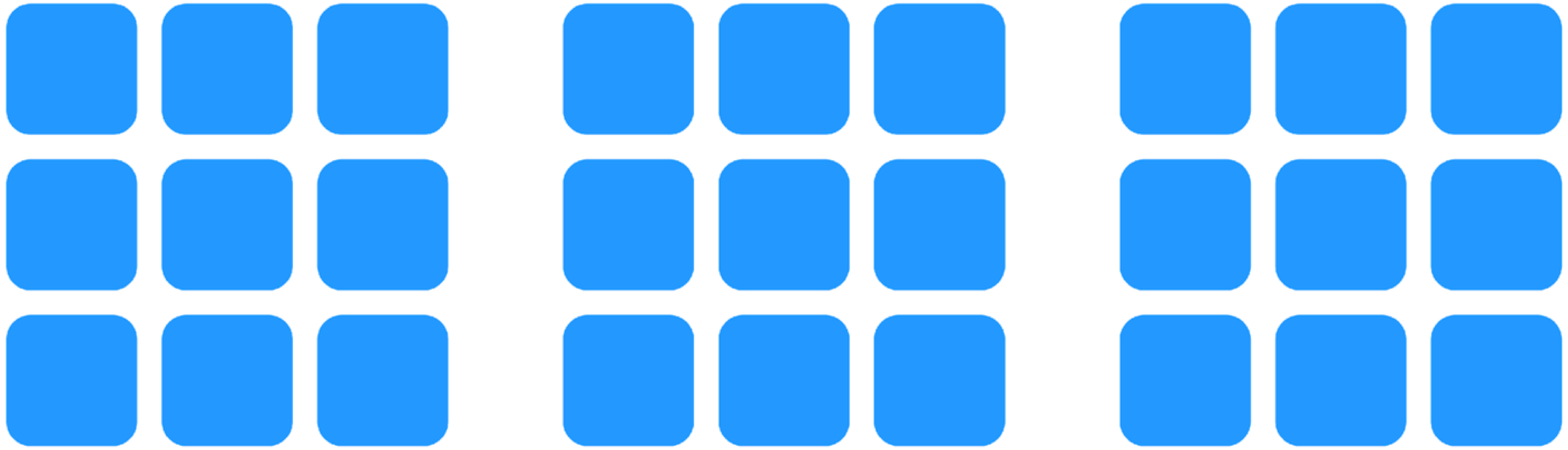
Nigel Poulton approves…

…and so do we

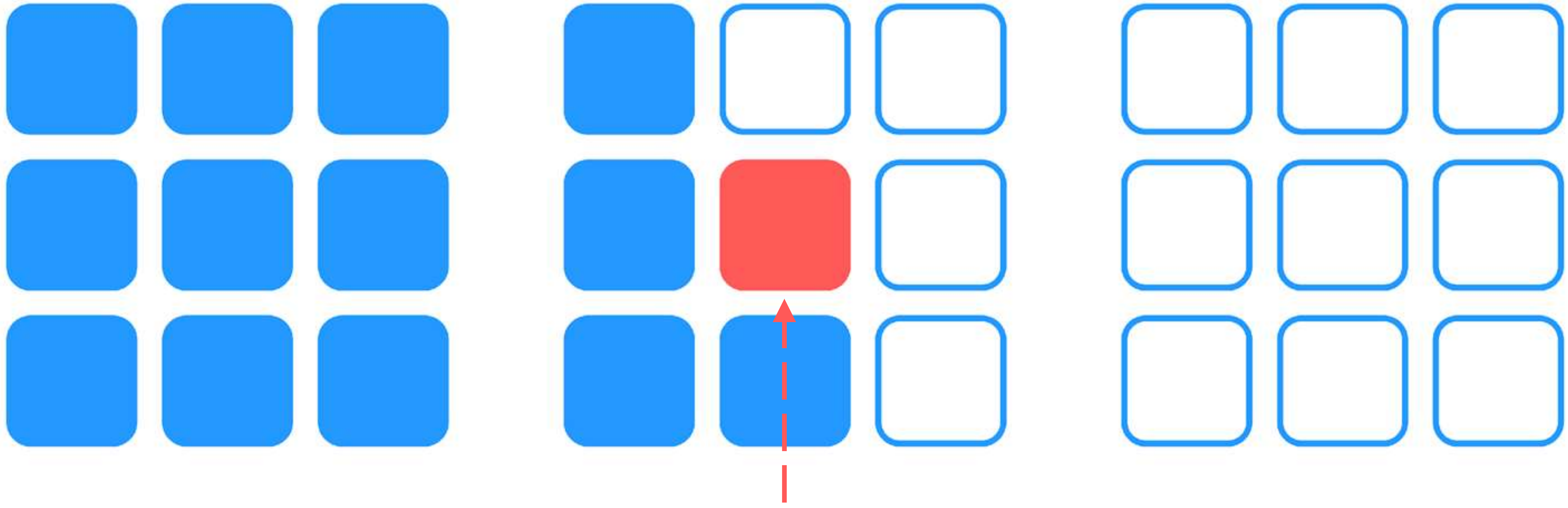
Quick reminder on operators



How to update the cluster’s version without operator?



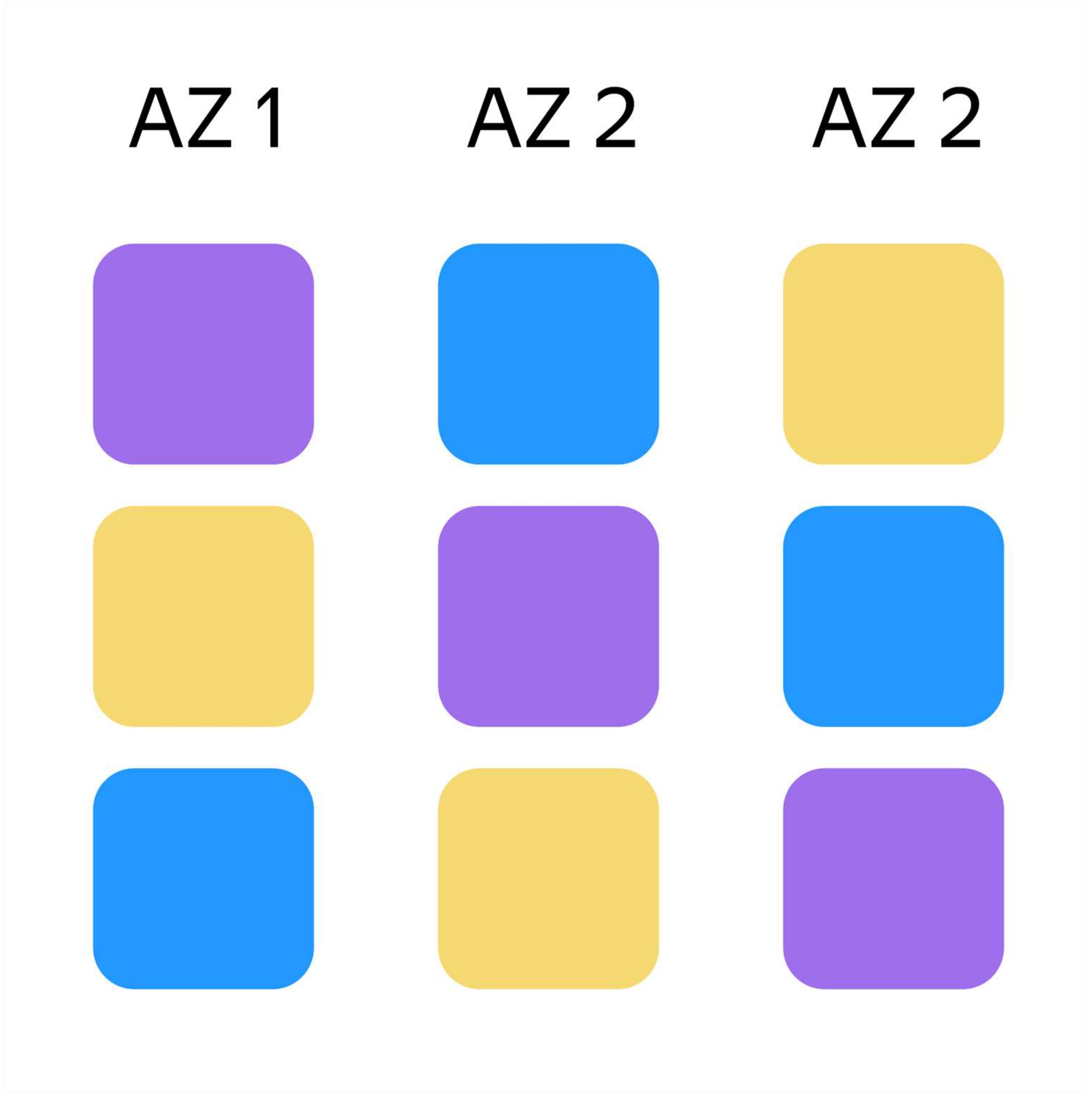
How to update the cluster's version without operator?



Power off the nodes one by one, the rest maintains high availability

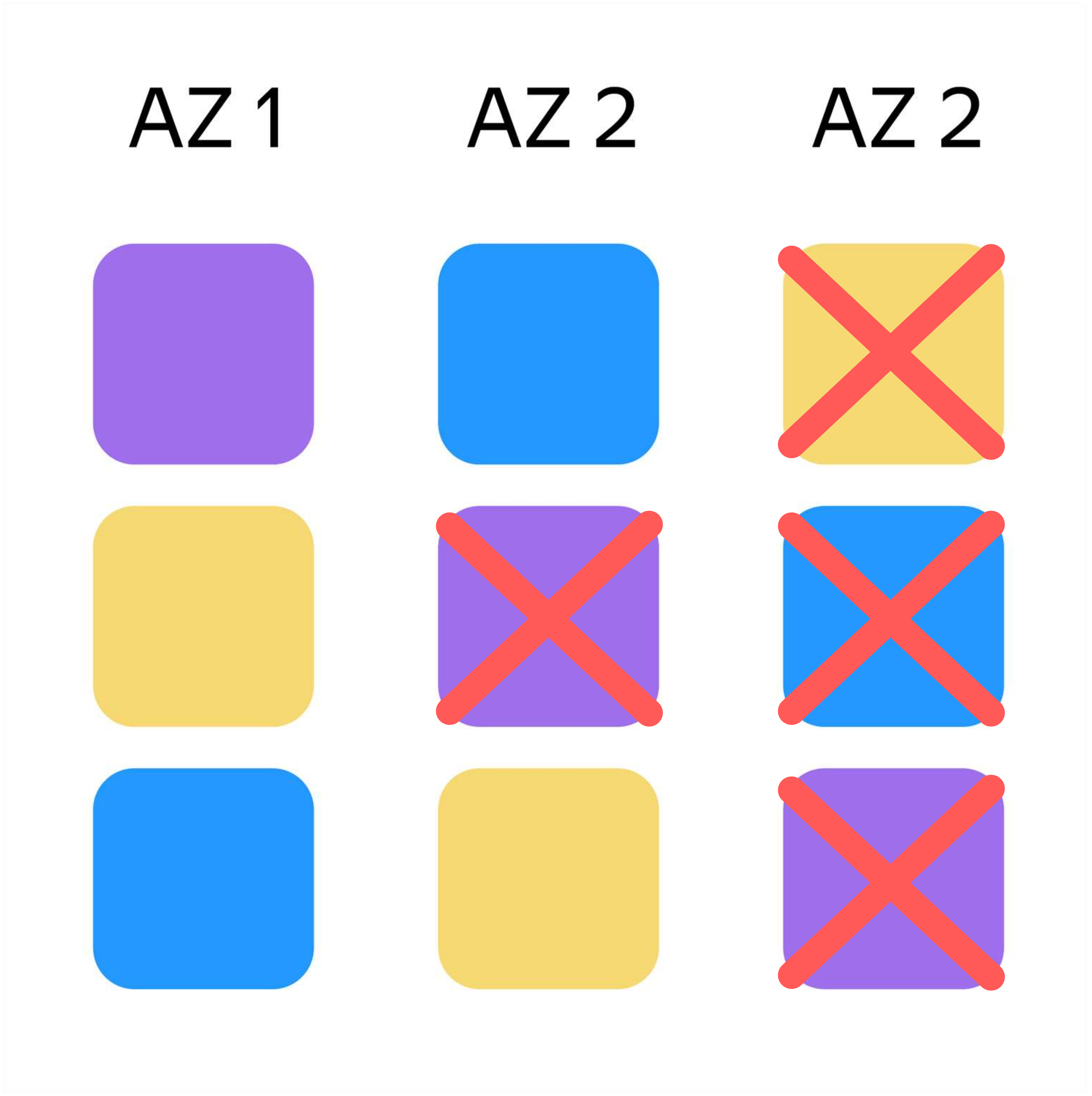
Some technical background

In YDB, data is split into logical groups. One group of data is replicated between availability zones:



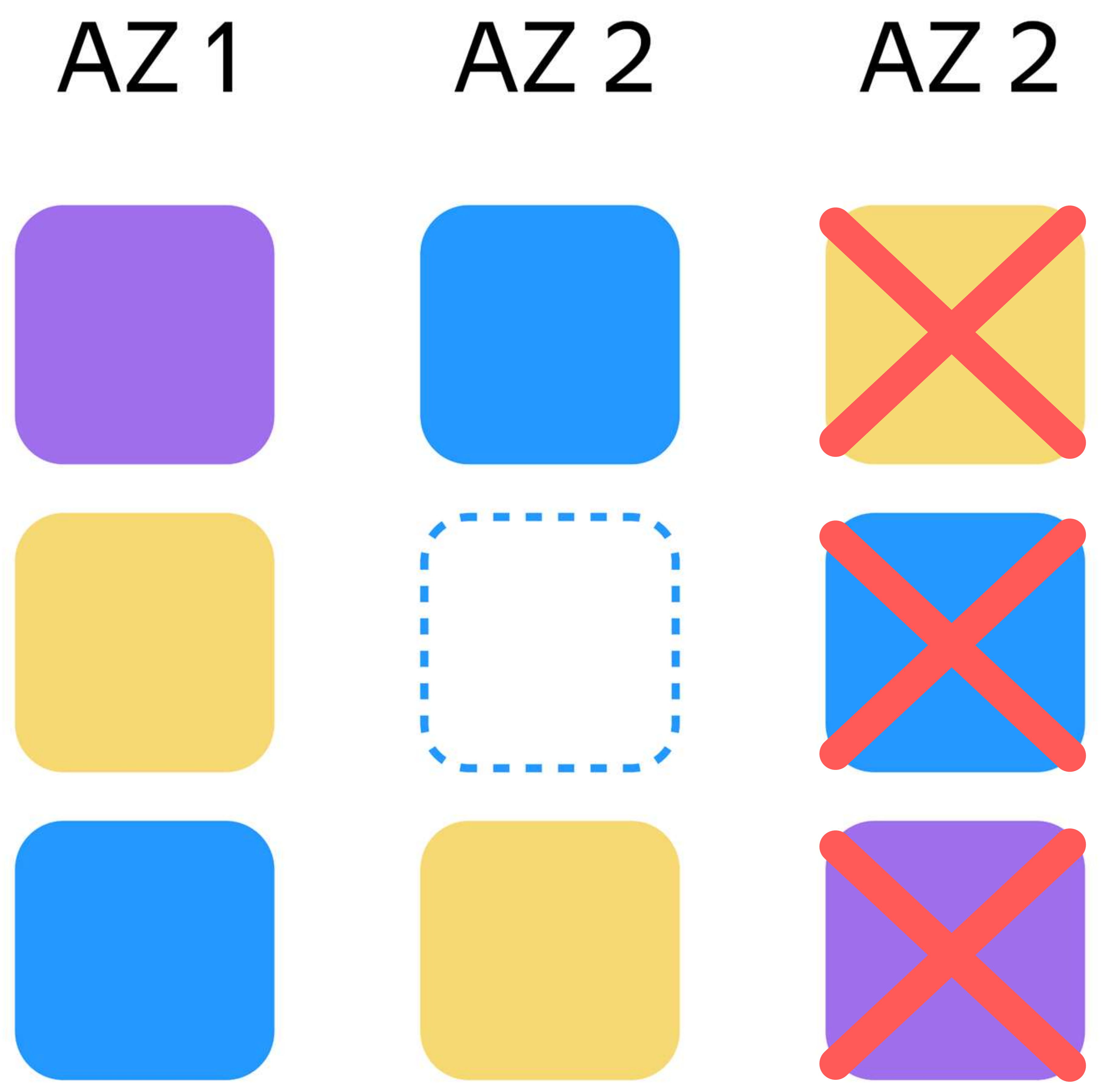
Some technical background

In YDB, data is split into logical groups. One group of data is replicated between availability zones, such that we survive the outage of an entire availability zone and another host:



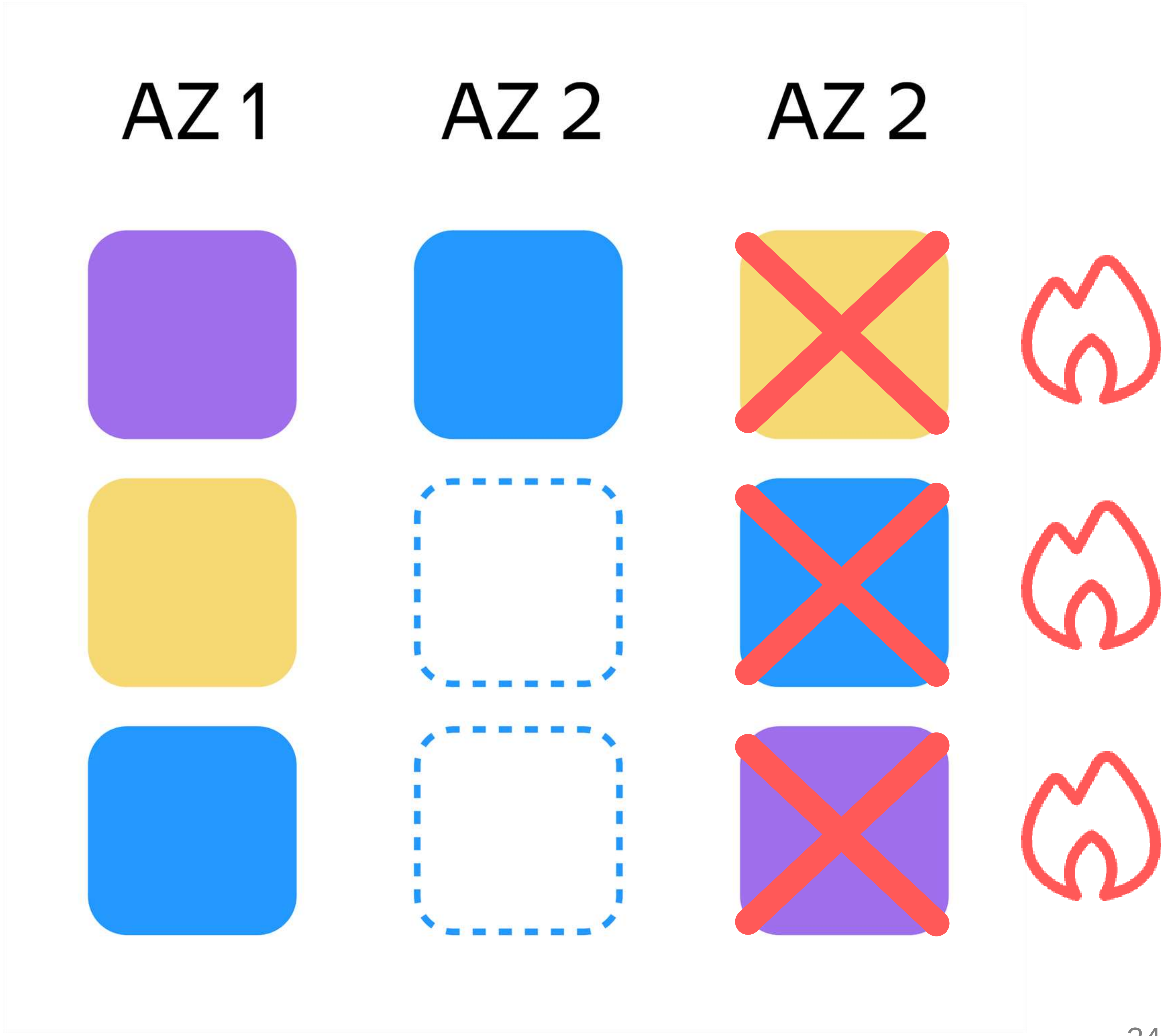
Maintenance reduces resilience

When we intentionally take nodes out, we reduce the high availability factor of the group (i.e. can survive less outage):

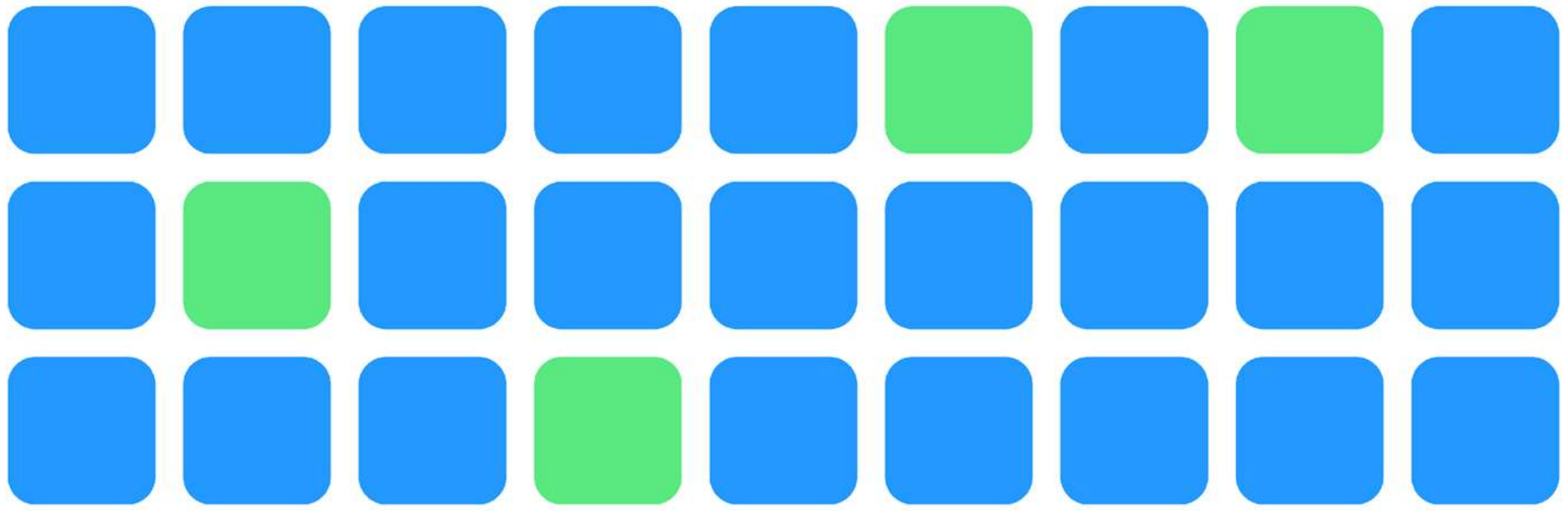


Maintenance reduces resilience

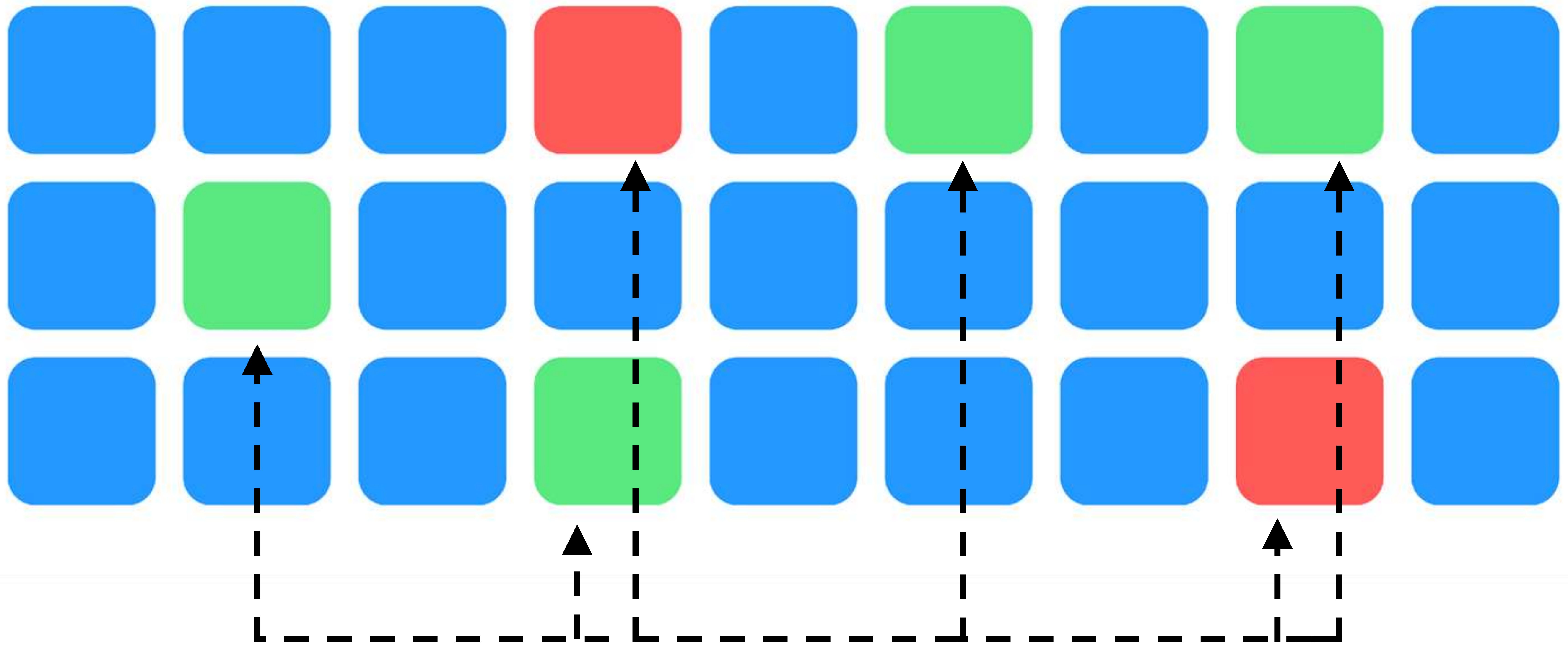
If we take a lot of nodes out at the same time, we can go accidentally go beyond our fault model



Re: how to update the cluster’s version with K8s?



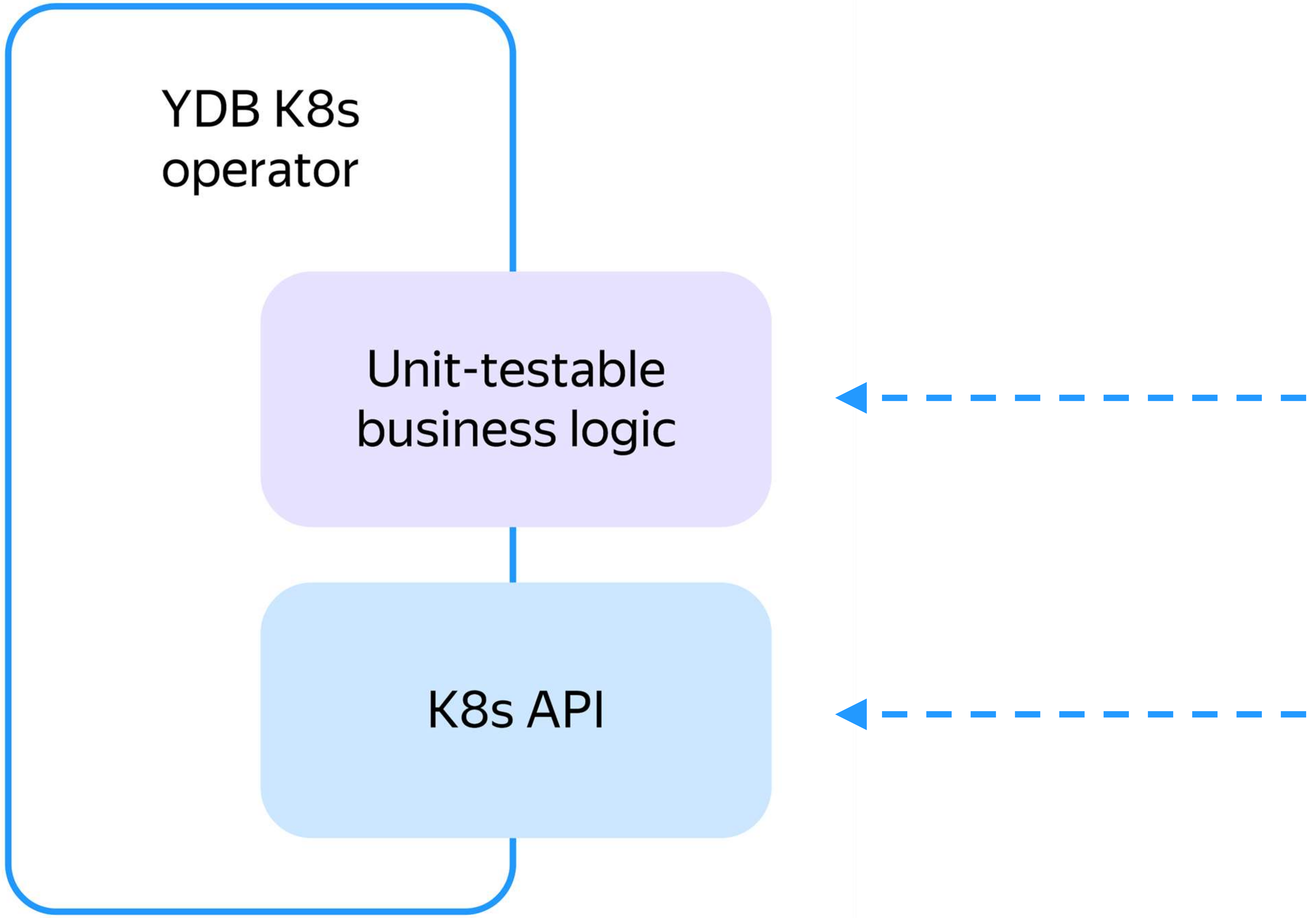
Re: how to update the cluster's version with K8s?



K8s operator: can I please take out those nodes?

YDB: okay, well, not all of them, but yes

Ecosystem aspect

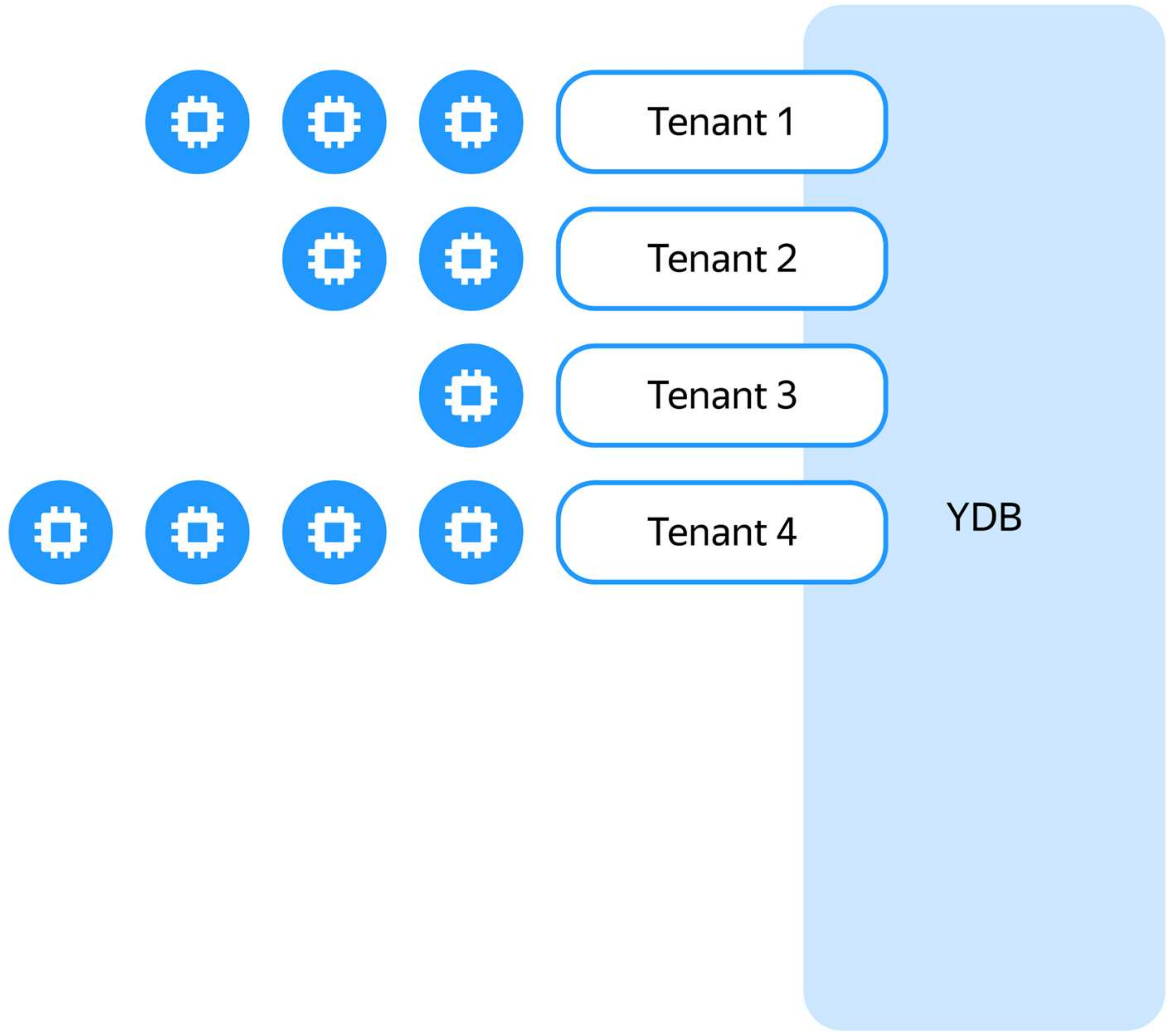


this is easily readable by the developer himself

this is well-known

YDB multitenancy

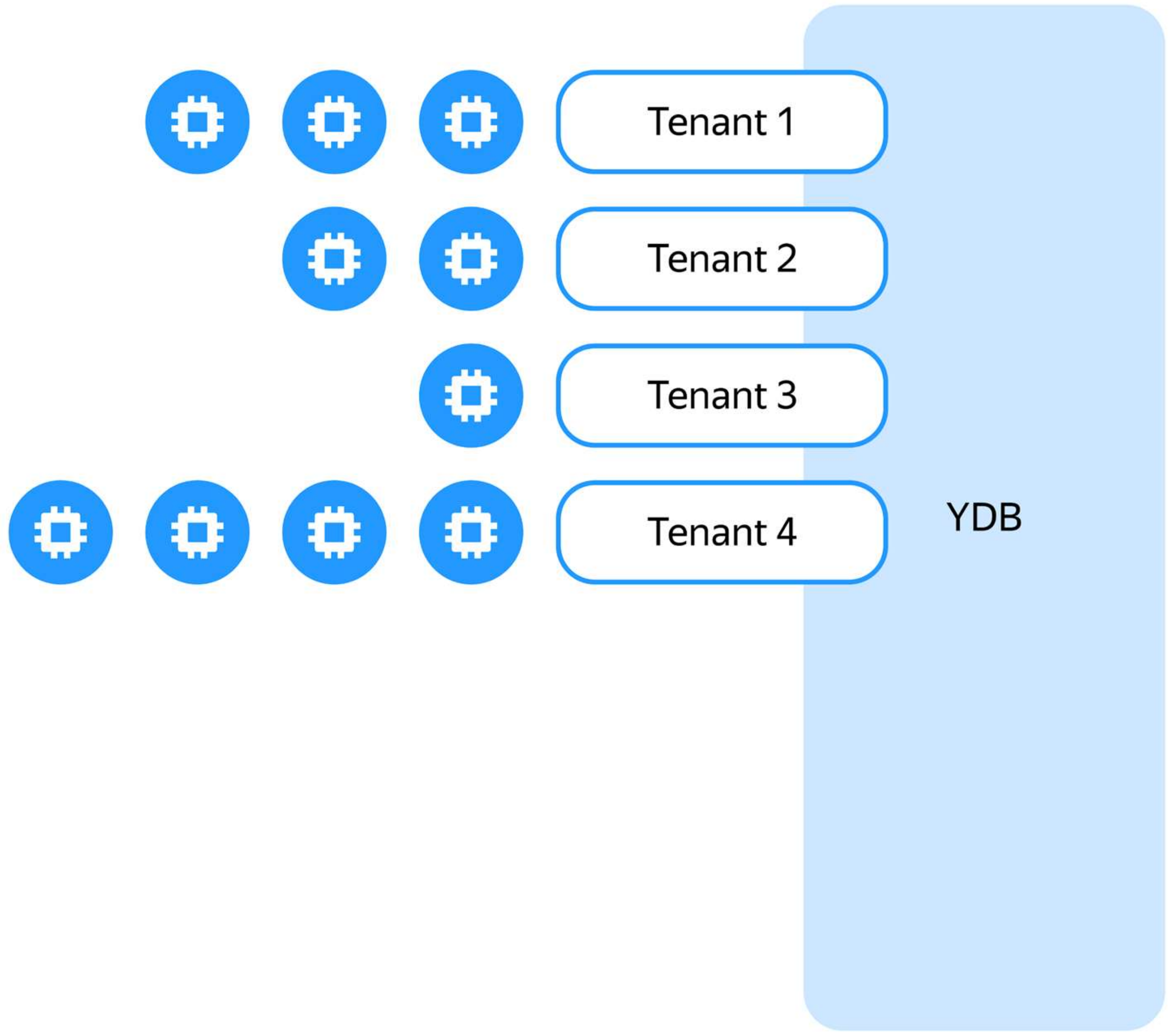
- Every cluster has multiple tenants
- Every tenant has its share of isolated resources
- It’s only natural to want to autoscale those individually!



YDB multitenancy

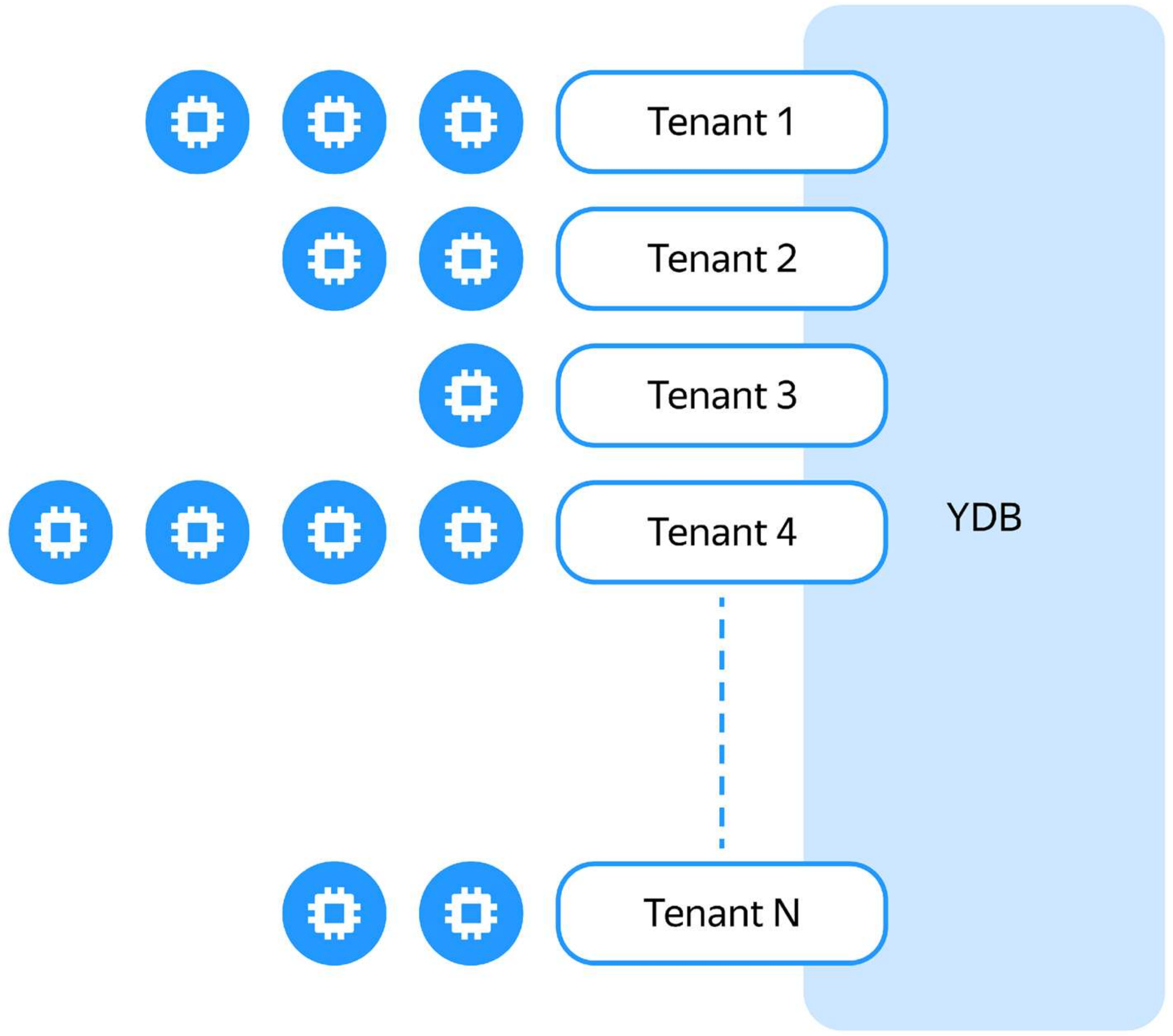
- Every cluster has multiple tenants
- Every tenant has its share of isolated resources
- It’s only natural to want to autoscale those individually!
- Let’s go Kubernetes native way, use Horizontal Autoscalers!

People with K8s background will come to this solution quickly enough



YDB multitenancy

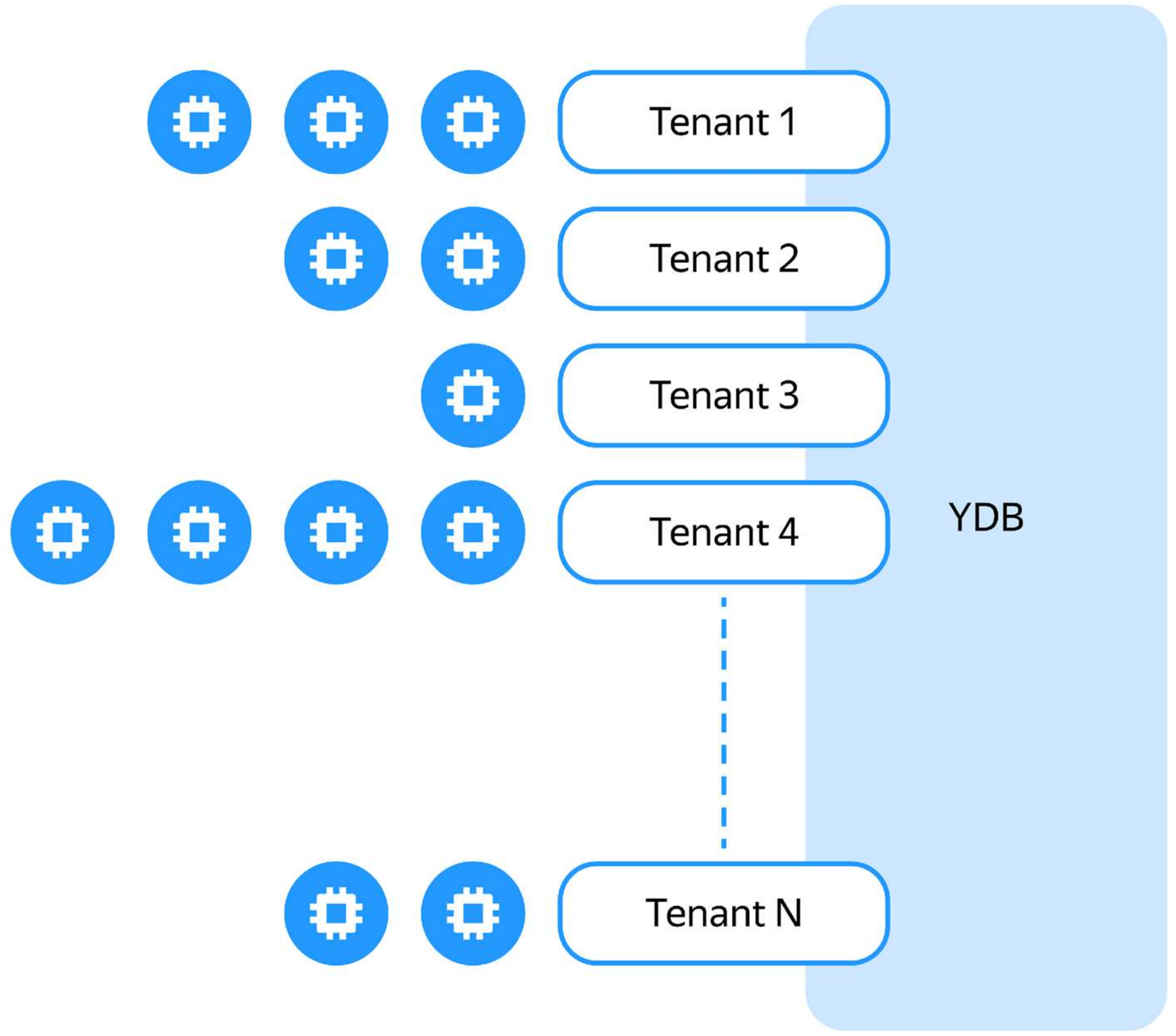
- But how do can we possibly know in advance, how many tenants are there at runtime?
- Thus, we can not configure Horizontal Autoscalers statically



YDB multitenancy

- But how do can we possibly know in advance, how many tenants are there at runtime?
- Thus, we can not configure Horizontal Autoscalers statically

But the operator knows that, and can create and update Autoscalers at runtime!



How do we work with drives?

One of our design decisions
is that YDB works with
raw block devices directly,
without operating through
a filesystem. Why?
Performance

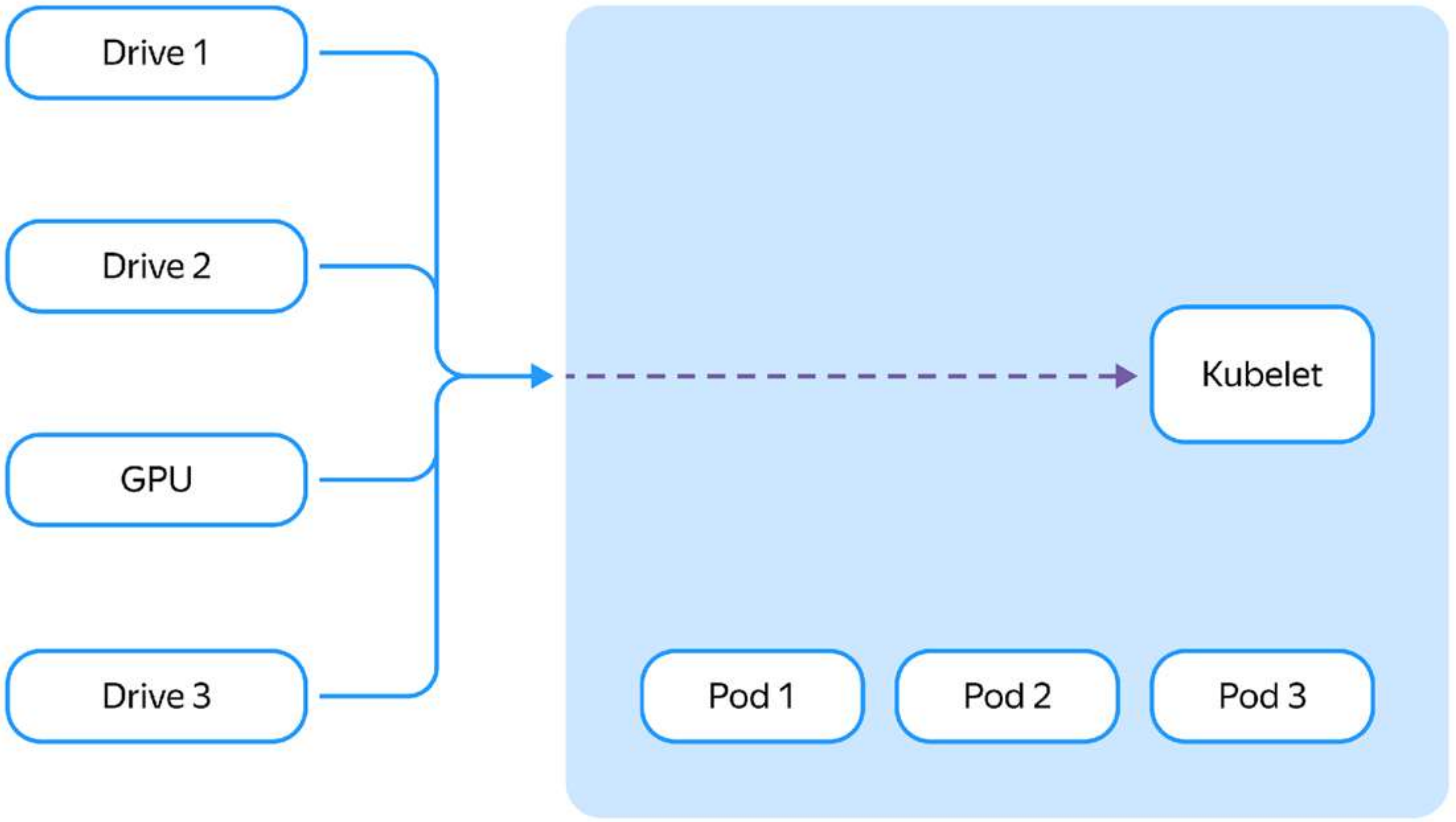
How do we work with drives?

One of our design decisions is that YDB works with raw block devices directly, without operating through a filesystem. Why? **Performance**

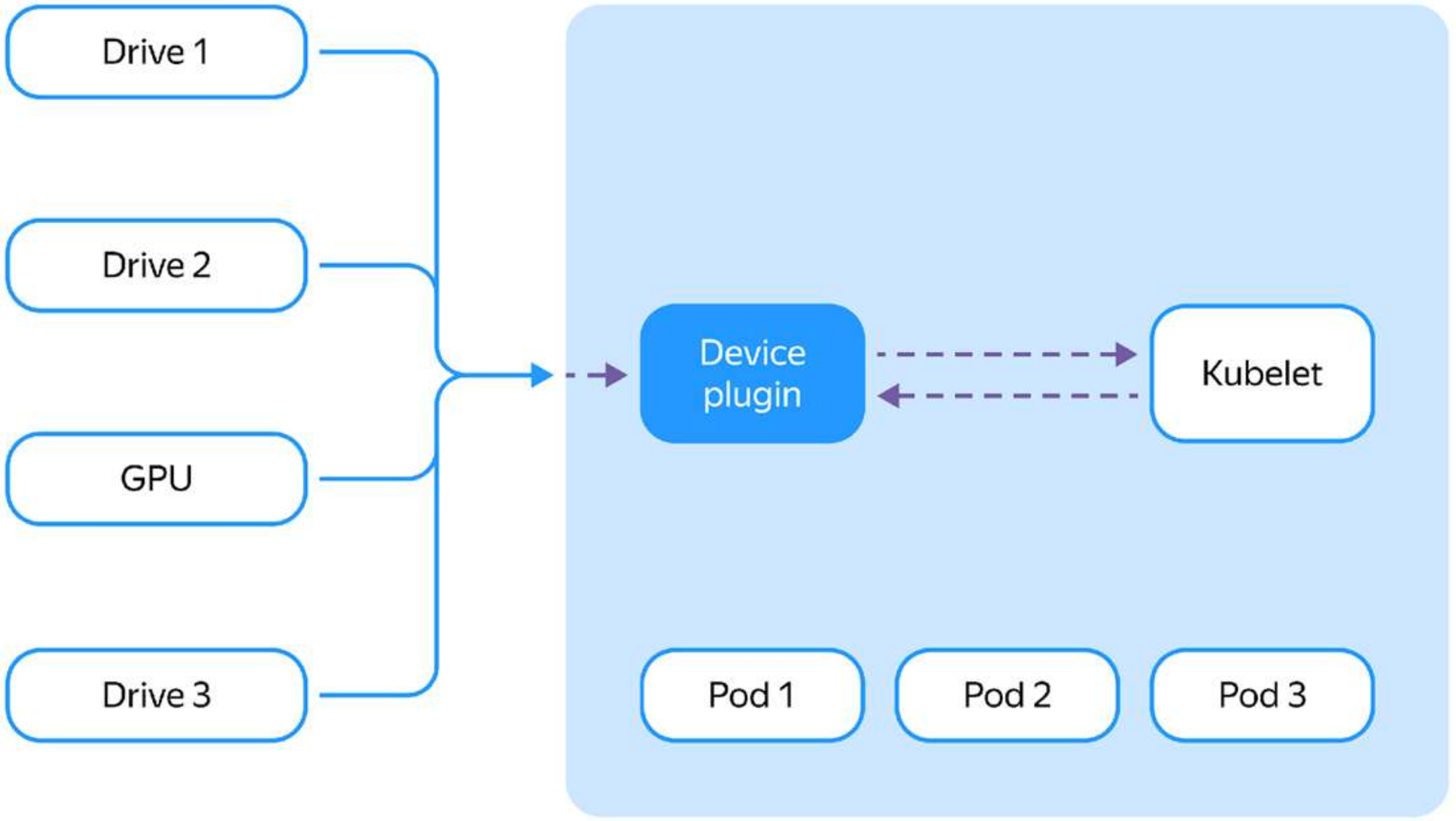
Downside: this requirement is really uncommon, so Kubernetes didn’t provide much tooling

Initially, we only could mount such a raw device into the Pod, if we allowed **superuser privileges** to the pod

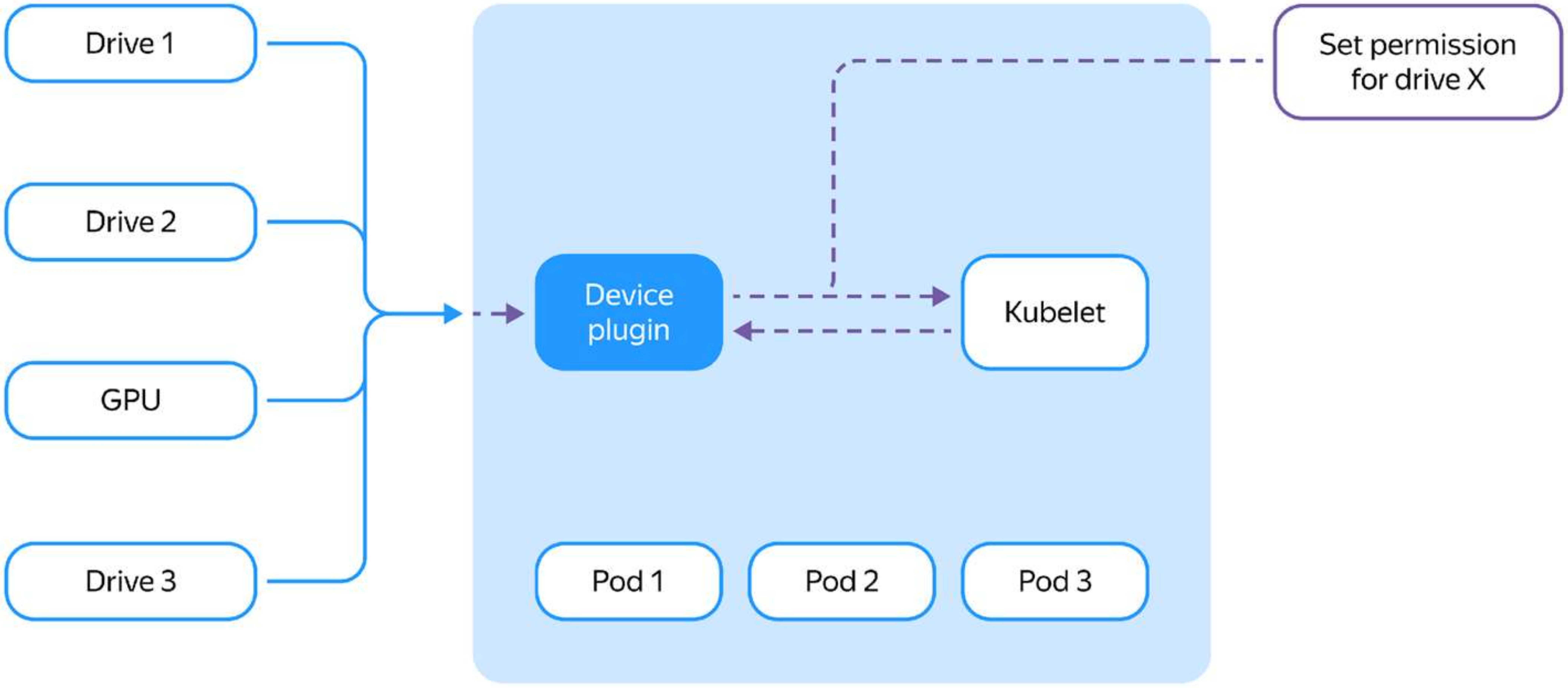
kubelet device plugin



kubelet device plugin



kubelet device plugin



What value did we obtain?

1

Autoscale

Tenant autoscalers

2

Ecosystem

Efficient onboarding

3

Flexibility

K8s does not get in our way

What is the moral of the story?

- Don’t be afraid of experimenting in general, and with Kubernetes in particular
- While we are on the topic of experiments… maybe YDB is for you?
- Booth number - G90

ydb.tech