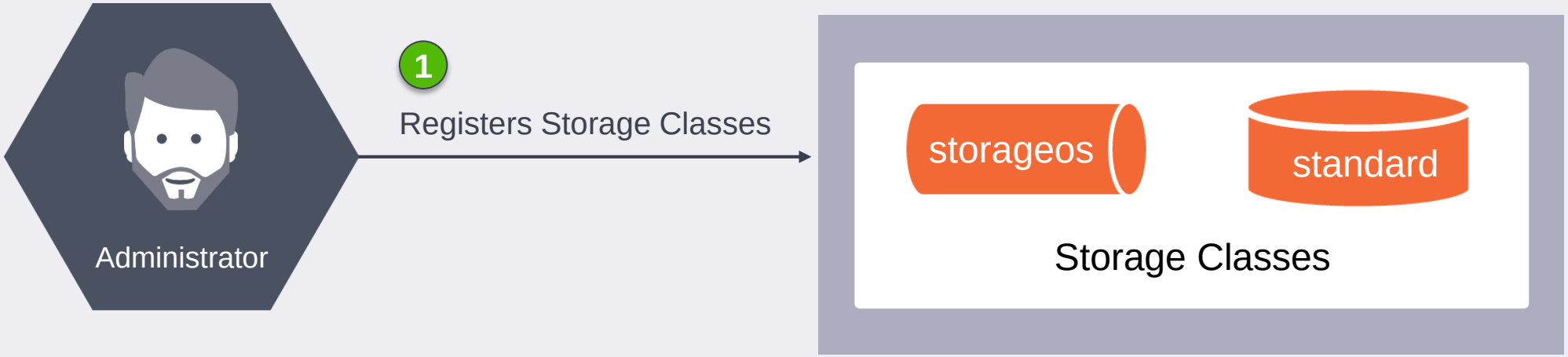
STORAGEOS

RANCHER®

Stateful Apps in Kubernetes:
Fast Database Recovery

Storage Class



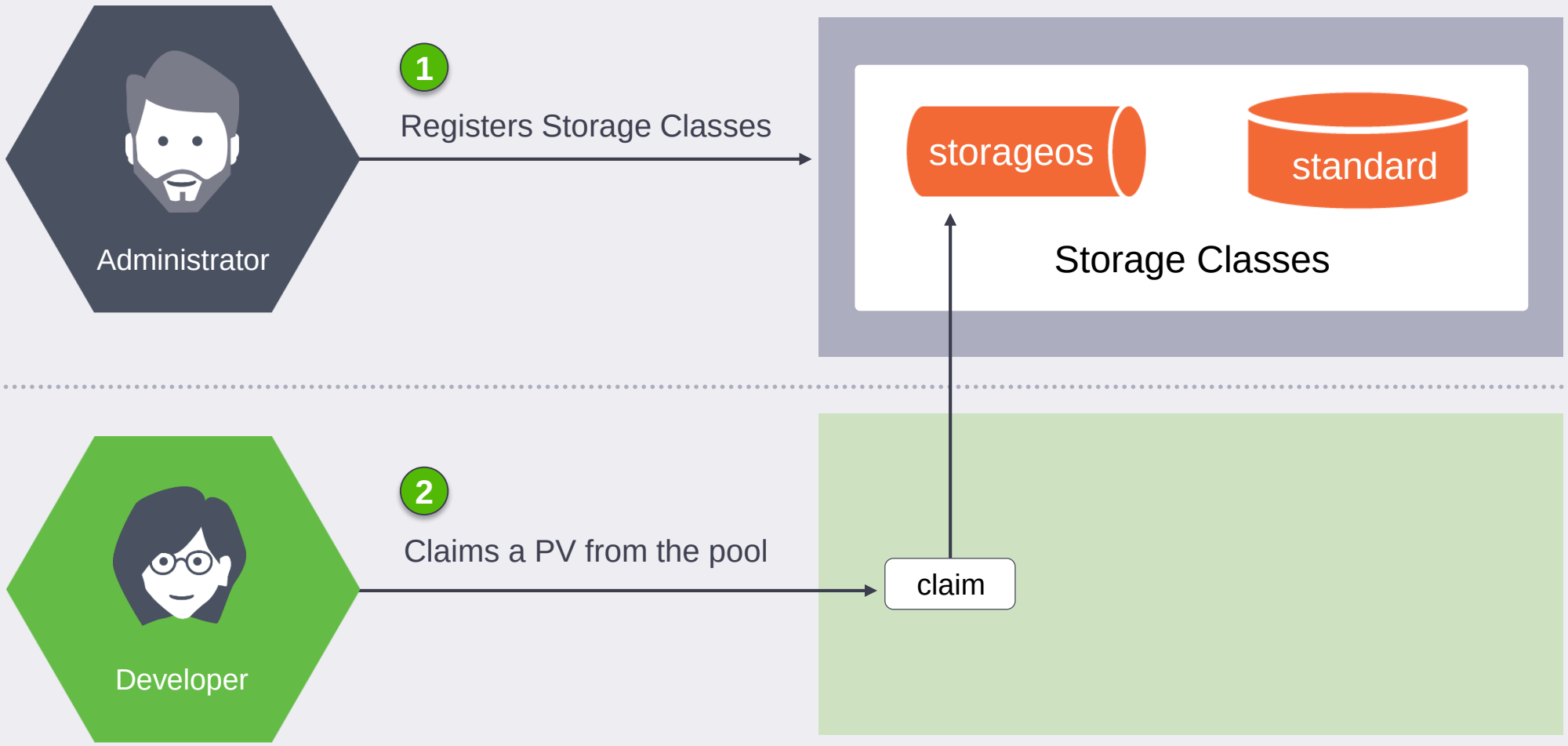
© StorageOS, Inc.

Storage Class

```
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
      name: storageos-production
      parameters:
        fsType: ext4
        pool: default
        storageos.com/replicas: “2”
        storageos.com/encryption: “true”
provisioner: storageos
```

© StorageOS, Inc.

Persistent Volume Claim



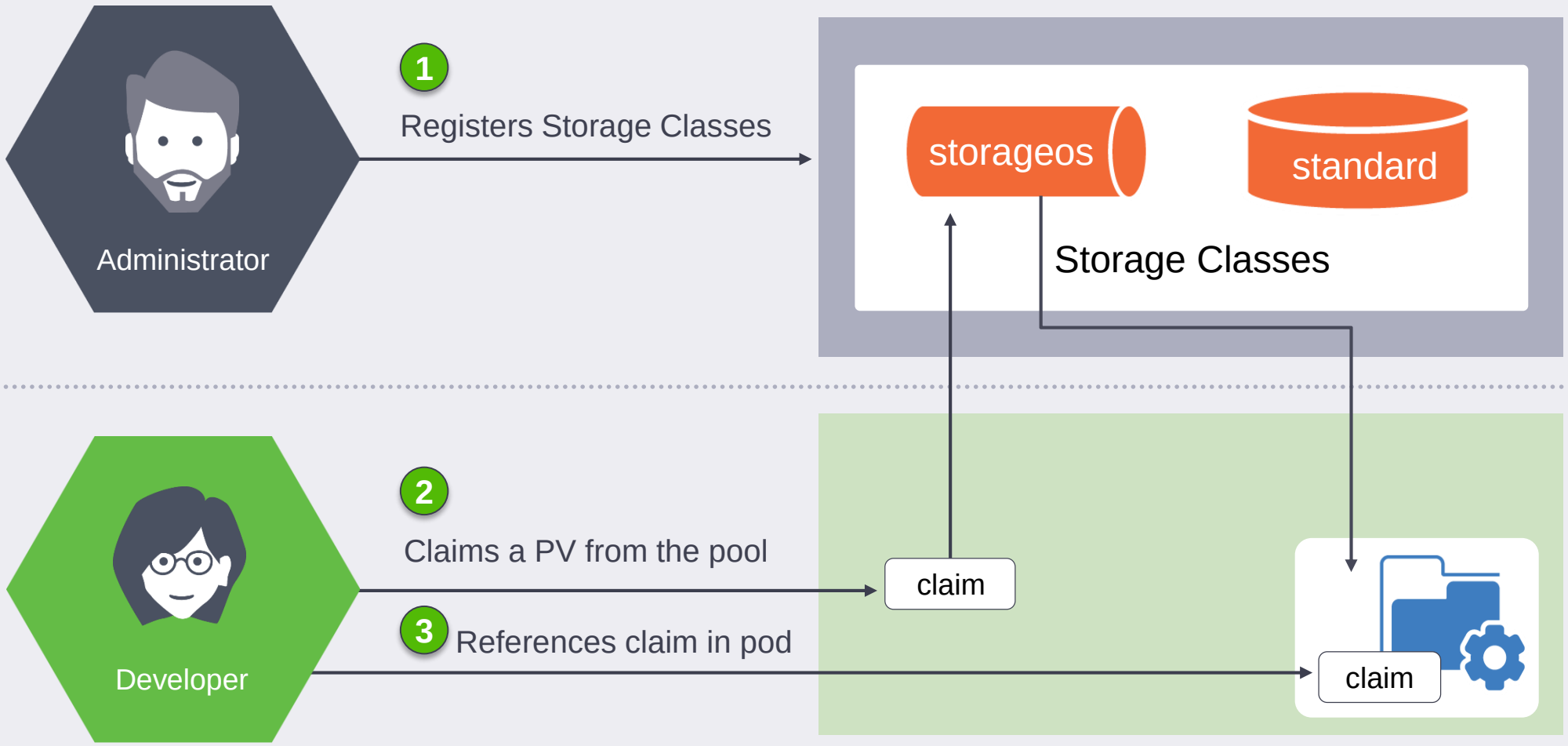
© StorageOS, Inc.

Persistent Volume Claim – Dynamic provisioning

```
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: database-vol-1
  annotations:
    volume.beta.kubernetes.io/storage-class: storageos-production
  spec:
    accessModes:
      - ReadWriteOnce
  resources:
    requests:
      storage: 100Gi
```

© StorageOS, Inc.

Persistent Volume Claim – Dynamic provisioning



© StorageOS, Inc.

STORAGEOS

Software-Defined, Cloud Native Storage

Provides persistent storage for containers

Gives you total control of your storage

Built from the ground up

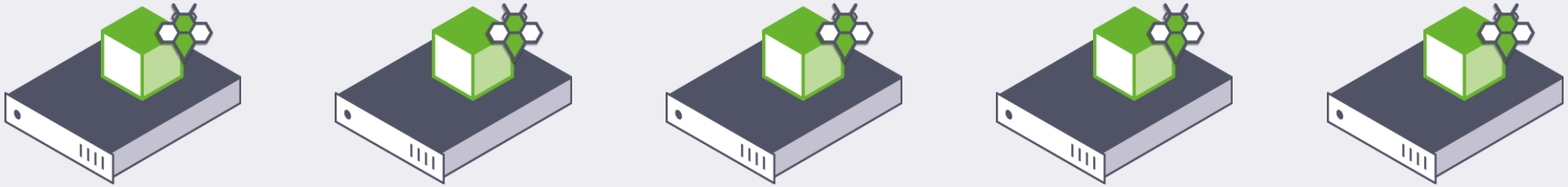
Highly performant container storage with features for today's digital business

Tech outcomes:

- Offer self-service storage
- In-memory caching and compression
- Synchronous replication and automatic failover for HA
- Use platform agnostic encryption to improve data security without performance overhead

© StorageOS, Inc.

Install StorageOS



© StorageOS, Inc.

Inside the StorageOS container

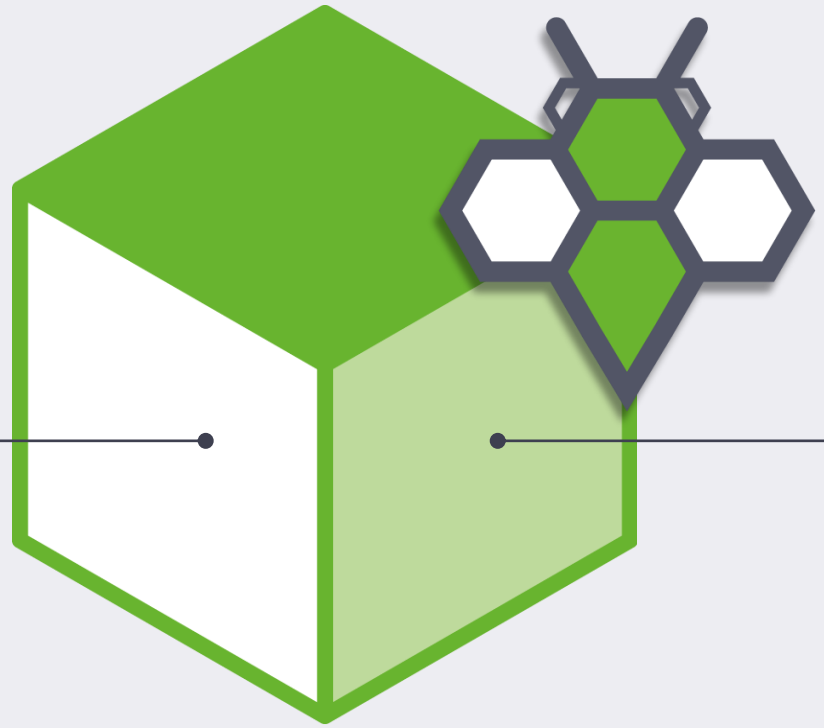
Control Plane
Intelligent

Volume placement

Detects node failure

Orchestrates cluster Operation

Kubernetes Communication



Data Plane
Performant storage engine

Hot path for all data

Replication

Compression

Encryption

© StorageOS, Inc.

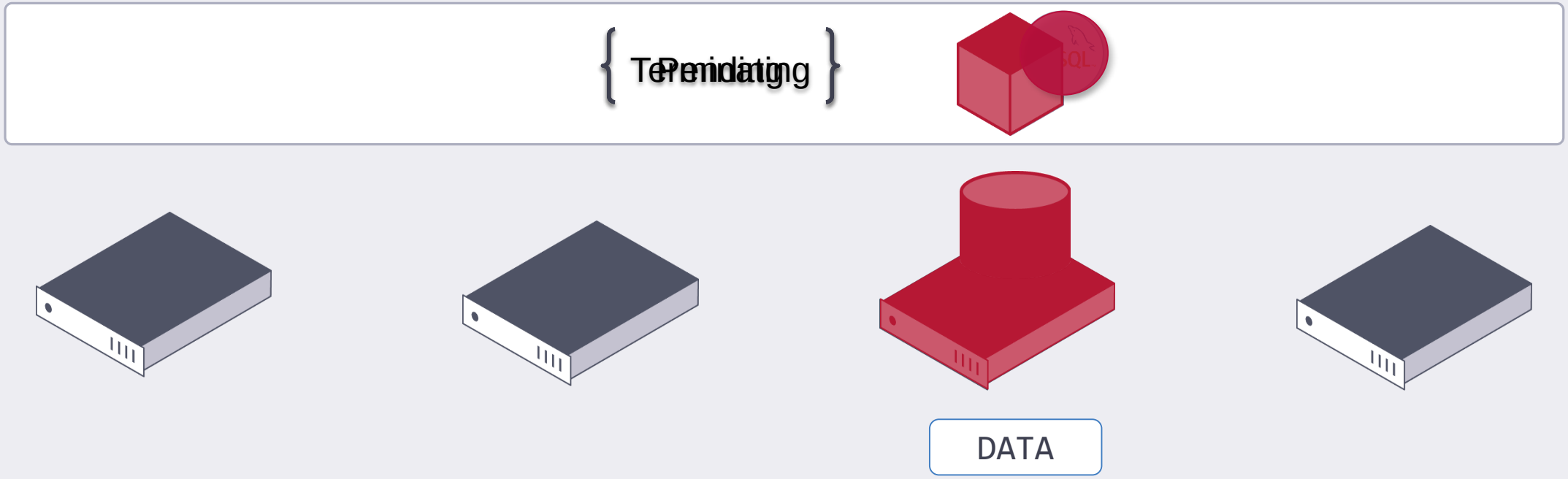
StatefulSet Behavior

Can’t access data

Can’t finish the Pod

Can’t reschedule Pod

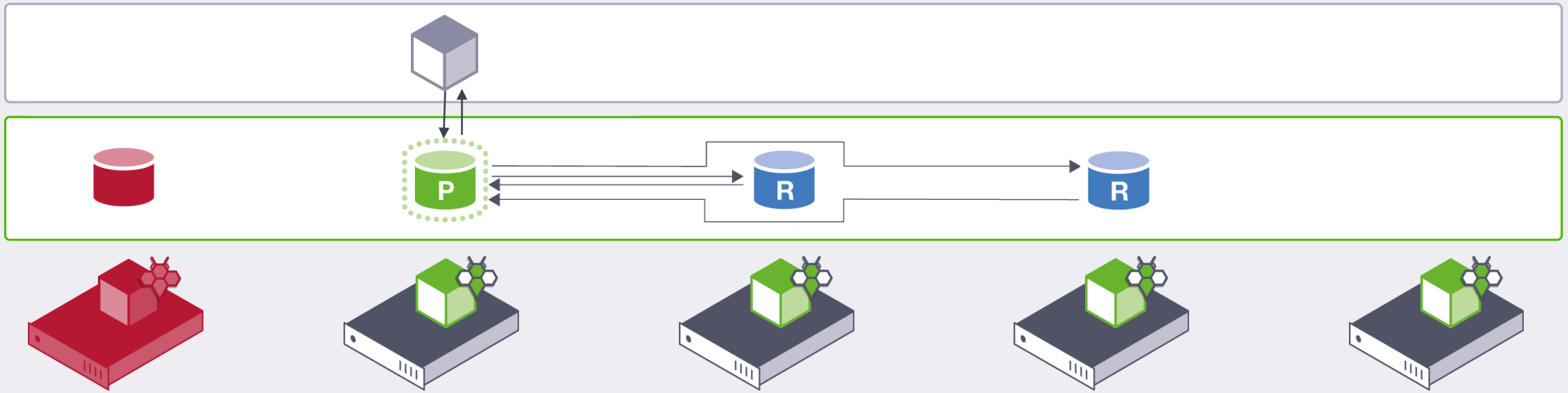
Service downtime



© StorageOS, Inc.

Feature: Replication for high availability

- Protects data from a disk or node failure and ensures a strong consistency model
- Replication is synchronous between a primary volume and user defined number of replicas (up to 5)
    1. Data is sent to the primary first
    2. Then sent in parallel to all replicas
    3. All acks need to be received by the primary.
    4. The write is acknowledged to the application.
    5. If a node fails, a replica is automatically promoted to become a new master and another replica is provisioned on an available node.
    6. Volume mount points move transparently to the application



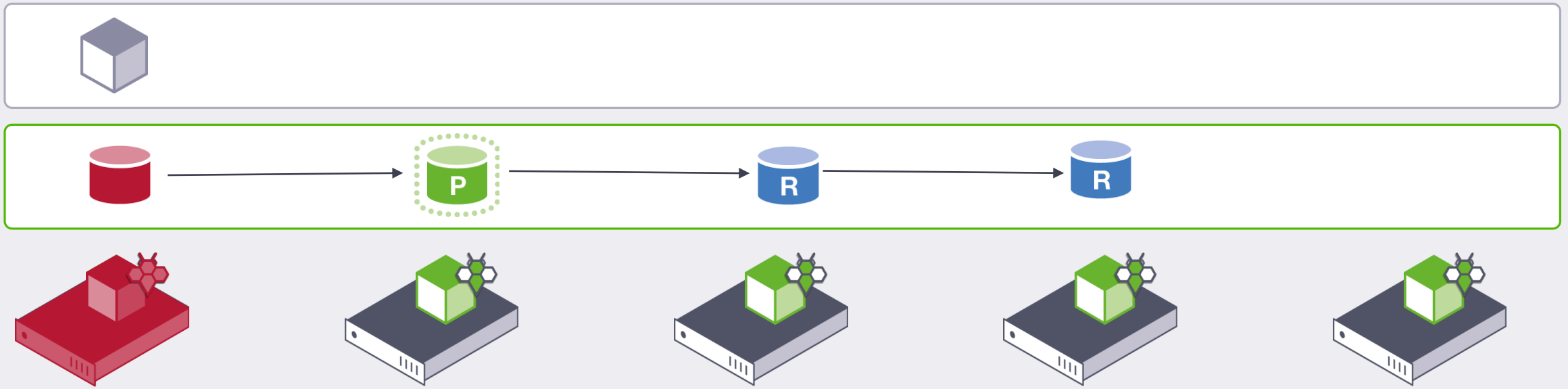
© StorageOS, Inc.

Feature: Rapid failover (fencing)

- StorageOS Rapid Failover uses StorageOS' awareness of node health to influence StatefulSet pod failover
  - When a node is dead, and StorageOS has scheduled a volume to failover, automatically kills the consuming pod and forces it to restart elsewhere in the cluster (when enabled)

Benefit

- Provides faster failover behavior than the StatefulSet controller alone



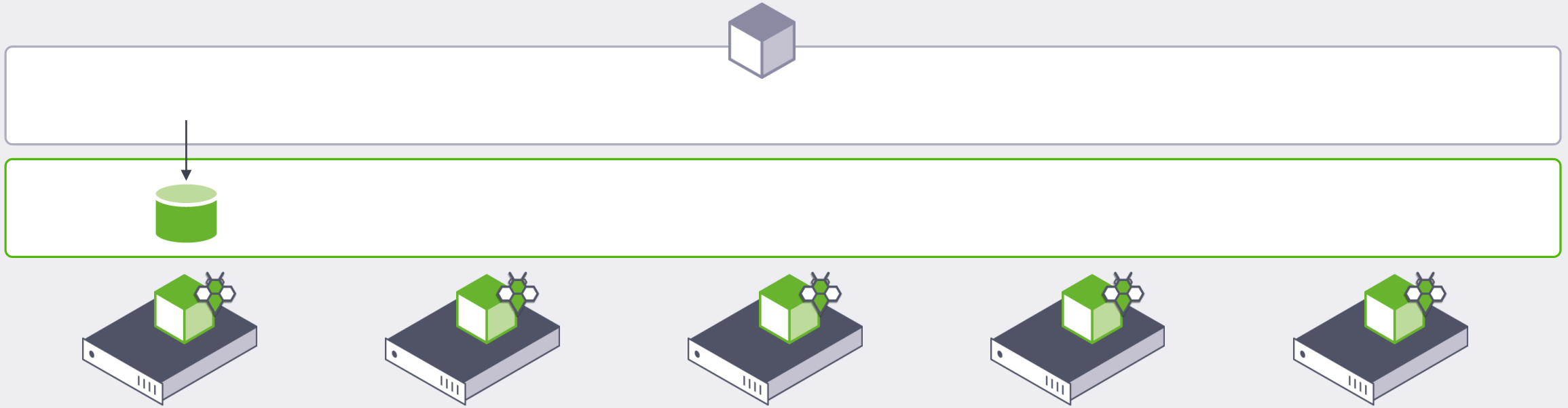
© StorageOS, Inc.

Feature: Locality

- StorageOS locality influences the Kubernetes scheduler to put applications next to the volume or move the volume next to the application
  - Where there is room on the node containing the master volume, place the pod there
  - When there isn't room on the master, but there is room on a node containing a replica, place the pod there and promote the replica
- Ensure workloads are collocated with volume

Benefit

- Collocate applications with the volumes they consume avoiding network roundtrips for performance sensitive workloads



© StorageOS, Inc.

Recap

1. Data must be accessible from the application, wherever is running
2. Data needs to be replicated
3. System needs to self heal automatically
4. StatefulSets can’t differentiate between failure and node isolation

   (Unless using StorageOS fencing, service downtime is guaranteed)
5. System needs to tolerate losing nodes
6. Apps follow storage when using StorageOS scheduler extension

STORAGEOS

RANCHER

Questions

Thank you!

Name: Ferran Arau Castell
Email: ferran.castell@storageos.com

www.storageos.com

© StorageOS, Inc.