KubeCon | CloudNativeCon

North America 2019

KubeCon | CloudNativeCon
North America 2019

Beyond Storage Management

Andrew Large and Yinan Li @Google

Agenda

- Data protection overview
- Data protection for Kubernetes
- Considerations

Data Protection Overview

- Goals
- Key Principles
- Approaches
- Policy
- Roles

Data Protection - Goals

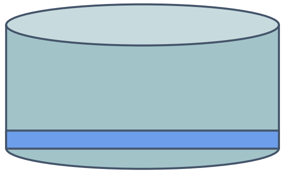
- To save a “point in time” state of the system to be used at a later time:
    - Recovery after failure
    - Workload/data cloning, replication, or migration
    - Offline data analysis
    - Pre-deployment testing
- Generally applies to two forms of “state”:
    - System configuration (e.g., host config, application installation and config, etc.)
    - Persistent data

Data Protection - Key Principles

- Recovery Point Objective (**RPO**)
  - Measure of how “out of date” (old) captured data is (lower is better)
- Recovery Time Objective (**RTO**)
  - Measure of how long it takes to recover from saved state (lower is better)

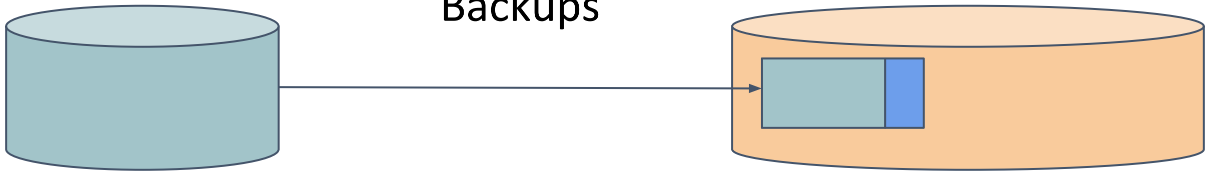
Data Protection - Approaches

Snapshots



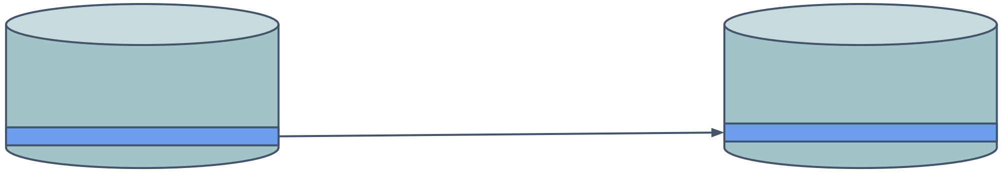
- stored inline (e.g., redirect on write, etc.)
- low RPO, moderate RTO
- moderate $$ (primary storage)
- low accessibility

Backups



- stored in different media (e.g., tape or object)
- moderate-high RPO, high RTO
- low $
- high accessibility

Replication (async)



- stored in same media, different location
- low RPO, low RTO
- high $$$ (primary storage * 2)
- moderate accessibility

Data Protection - Policy

- Considerations
    - Simultaneously minimize RTO/RPO and $
- Common approach - mix of snapshots and backups
    - Small number of snapshots to minimize RTO/RPO
    - Larger number of backups to cover additional use cases
    - Scheduled snapshots and backups with expiry/deletion
- 3-2-1 rule
    - Keep at least 3 copies of your data
    - Store 2 backup copies on different devices or storage media
    - Keep at least 1 backup offsite

Data Protection - Roles

- Infrastructure Administrators
  - Setup and manage infrastructure
  - Have full access to systems
  - Execute data protection policy
  - May not have detailed understanding of workloads
- Application Administrators
  - Install, upgrade, and manage applications
  - Restricted/delegated access to system
  - Have detailed understanding of workloads

Data Protection for Kubernetes

- Scope
- Active Efforts
- Potential Future Efforts

Data Protection for K8s - Scope

- Configuration
    - “GitOps” - treat config as code and manage/deploy from source code control
    - Backup/Recovery - treat config as state and perform regular backups (using backups for recovery)
    - Hybrid - GitOps for cluster resources, backup+recovery for applications
- Data (in PersistentVolumes):
    - Volume snapshots - stored in the local cluster storage pool
    - Volume backups - stored outside the local cluster (typically in object storage)

Data Protection for K8s - Active Efforts

- Volume Snapshots
  - Uses Custom Resource Definitions (CRDs), enhances Container Storage Interface (CSI), and new CSI driver sidecar
  - Alpha in 1.12
  - Beta targeted for 1.17

Data Protection for K8s - Potential Future Efforts

- “Plugin” PVC data populators
  - Existing PVC “dataSource” is difficult to evolve
- Volume backups
  - With explicit extra and inter-cluster semantics
- Volume groups (consistency groups)
  - Purpose: capture a single “point in time” across multiple volumes
  - Challenge: models vary widely between storage vendors
- Application-consistent snapshot/backup
  - Point-in-time capture of a running application, including app config and persistent data

Considerations

- Volume backups
- Layered administration
- Application consistency
- Application awareness
- Application-mediated backup

Considerations - Volume Backups

- Existing volume snapshots:
    - Backup-related semantics too unclear for portable data protection policies
    - Missing target location
    - Missing global ID or defined import/export flow
    - Tightly coupled with primary storage
- Multiple backup models desirable:
    - Provided by primary storage (if supported)
    - Provided by separate backup provider (allows for backups that are portable between storage systems)

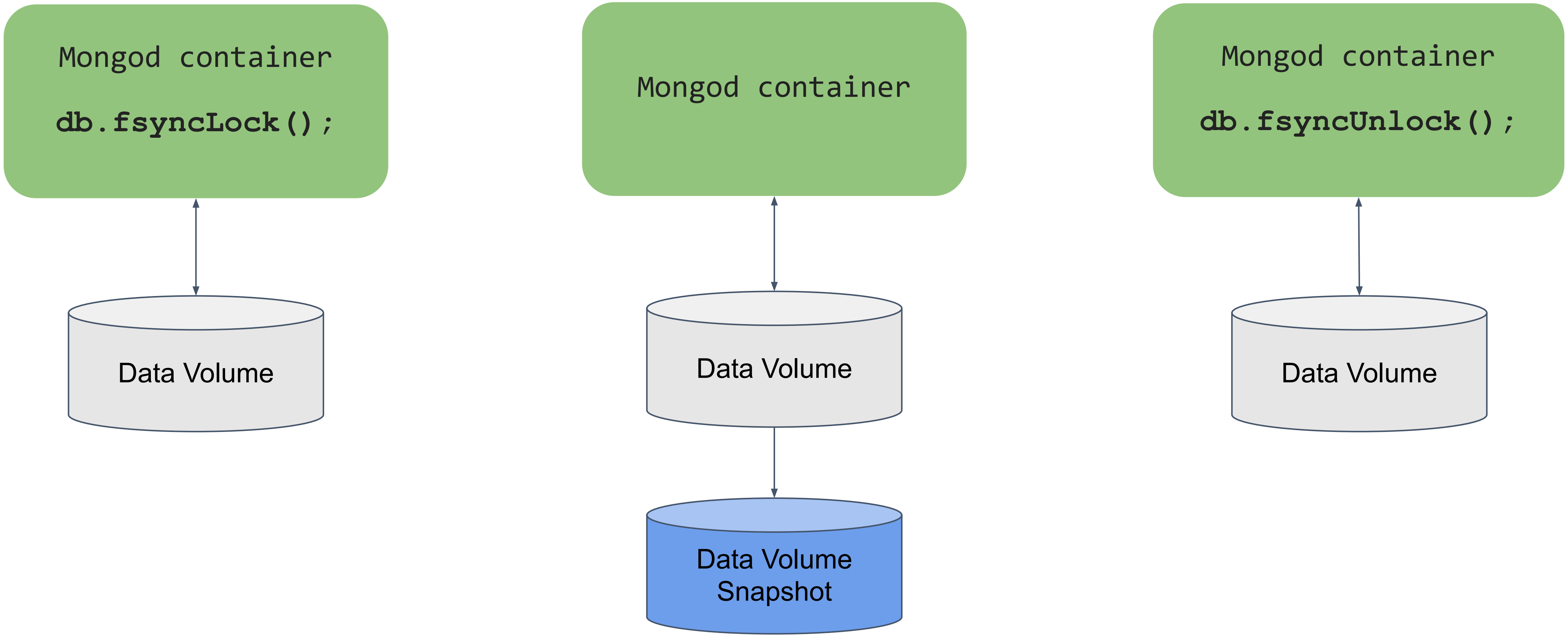
Considerations - Layered Administration

- Issue - infrastructure administrators may not know how to orchestrate application backups
- Approaches:
  - Rely only on generic hooks (e.g., “fsfreeze”)
  - Treat application backup and recovery as a separate problem from cluster backup/recovery
  - Provide some mechanism to automatically orchestrate application backups as part of cluster backup

Considerations - Application Consistency

- Goal - ensure that an entire application’s state is recoverable
  - Typically involves a “flush” and “quiesce” step before capturing volume data and an “unquiesce” step afterwards
  - Generally required only when application has multiple volumes or doesn’t maintain crash-consistency of persistent data
- Windows has VSS - no equivalent for Linux/K8s
  - Common Linux/K8s approach is to define “hooks” which run commands inside containers
  - Hooks may be generic (e.g., “fsfreeze”), but application-specific commands are also likely

Considerations - Application Consistency


Mongod container

`db.fsyncLock();`

Mongod container

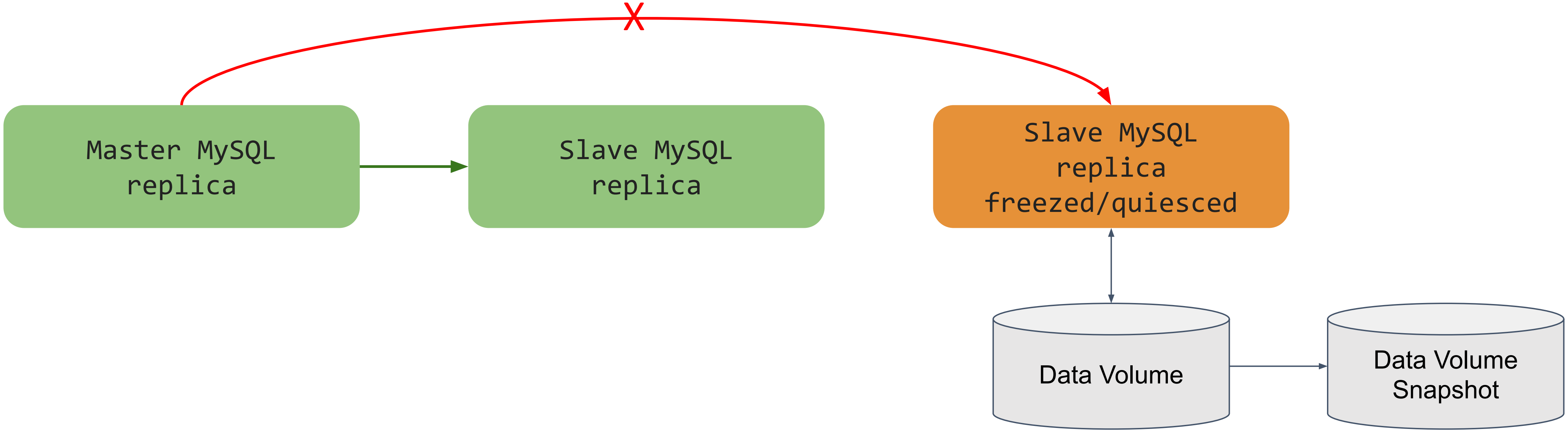
Mongod container

`db.fsyncUnlock();`

Considerations - Application Awareness

- Goal - smart application-aware orchestration
  - Backup orchestration takes advantage of deployment architectures to avoid downtime during backup
- Example orchestration
  - Finding and picking a secondary replica
  - Take that replica temporarily out of replication
  - Flush and quiesce that replica
  - Backup that replica’s volume(s)
  - Unquiesce and put it back into replication
  - Recover both primary and secondary replicas from same backed-up volume(s)

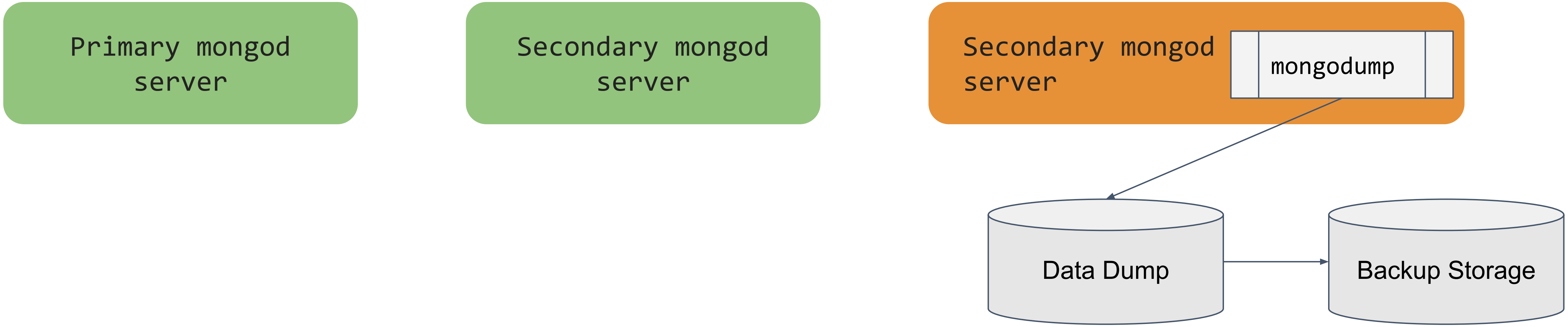
Considerations - Application Awareness



Considerations - Application-mediated Backup

- Goals - backup using application-specific tools and methods
  - Data portability
  - No down time, but likely with performance penalty
- Example orchestration
  - Find and pick a secondary replica
  - Run some tool against that replica to perform a data dump
  - Upon completion of the data dump, upload the dumped data files to backup storage
  - Use the dump data to restore all the replicas, again using some application-specific tool

Considerations - Application-mediated Backup



Summary

- Data protection on Kubernetes is a multi-persona concern
- Data protection on Kubernetes has a lot of potential use cases: disaster recovery, migration, safe upgrades, etc.
- Storage management in Kubernetes goes beyond bare volume snapshots
- Many considerations go into building a data protection system for Kubernetes

Questions?