Self-service Data Protection for Stateful Containers

Umesh Deshpande*, Nick Linck, Sangeetha Seshadri
IBM Research – Almaden, San Jose, CA
Contact*: udeshp@us.ibm.com
Data Protection in Containerized Environment

- **Rapid adoption of container native storage**: According to IDC, 90% of applications on cloud platforms and over 95% of new microservices are being deployed in containers.

- Users of containerized environment expect self-service model for data protection, like other services, e.g., fault tolerance, load balancing.
Challenges in Providing Data Protection Guarantee

**Recovery Point Objective (RPO):** The RPO is said to be $T$ hours if the application can lose no more data than the changes made in the last $T$ hours.

**Challenges**

- User may not know if the infrastructure can guarantee the specified RPO
- Applications and backups competing for resources

**Goal:** Self-service data protection to a large number of volumes with varying RPOs in face of resource outages and fluctuations
Data Protection with Volume-level Snapshotting

- Snapshot is a point-in-time representation of a volume

- **Incremental**: Only capture the changes since the previous snapshot

- **Quick**: Crash-consistent snapshots do not require state synchronization
Existing Work

- Periodic Backups for Containers
  - Velero, KastenIO, IBM Spectrum Protect Plus
- Backup Optimization
  - Reduce overhead
    - [Natanzon et.al., NAS’16]
    - [Cherkasova, et.al., MASCOT’09]
  - Selectively backup applications
    - [Kettimuthu et.al., SC’15]
- Quickly react to the failures or resource fluctuations for RPO compliance
Self-service Data Protection

- User need not dictate when or how often the volumes are snapshotted or backed up

- Resiliency against resource, component and backup job failures by treating each operation (request creation, snapshot, data copy) as a transaction

- **Adaptive scheduling** of backups to provide data protection for volumes with a wide range of RPOs

```yaml
apiVersion: "backup.io/v1"
kind: BackupReq
metadata:
  name: <request_name>
  namespace: <namespace_name>
spec:
  requesttype: Backup
  rpo: <time in minutes>
  retention: <time in minutes>
```

Backup Request using a Yaml configuration file.

User can observe status of the request with...
kubectl get backupreq <request_name>
Insight Behind Snapshot Scheduling

• **Insight**: Reducing the interval between snapshots can allow more time for data copying without RPO violation.

• Scheduler increases the snapshot frequency if the backups are falling behind, e.g., due to resource contention or outages.

• **Assumption**: $X_1 + X_2 < 2 * S_1$

RPO = 4 hours. Frequent snapshots capture smaller change and allow more time for copying out the data without RPO violation.
Adaptive Scheduling for Backups

Phase 1: Snapshot Scheduling

- **Snapshot Now?** = $F_n($Per-volume slack, Cluster slack$)$
- **Per-volumes slack**: Indicator of flexibility w.r.t. the amount of data, RPO and predicted bandwidth.
- **Cluster slack**: Indicator of how well the backups across the cluster are meeting their deadlines.

Phase 2: Data Copy Scheduling

- Snapshots with lower slacks are copied first
Evaluation and Test Setup

Simulate 2500 volumes with varying RPOs over a 10 day period

- **Rate of Change Models**
  - Uniform
  - Bursty
  - Variable

- **Bandwidth Models**
  - Outage
  - Spikes
  - Periodicity

**Comparison of Scheduling Strategies**
- 2 variations of fixed scheduling with different aggressiveness
- 2 variations of adaptive scheduling with different aggressiveness (with volume-level information)
- Cluster-aware Adaptive Scheduling

**Metrics**
- Percentage of time spent in RPO violation
- Number of snapshots
Observations:

- Frequent snapshots are helpful in reducing RPO violations
- **Timeliness**: Important to perform snapshots when necessary
Number of Snapshots

Observations:

- Adaptive approach reduces RPO violation with similar number of snapshots as the fixed-proactive approach.
Conclusions and Future Work

• To summarize,
  • **Self-service**: User need not dictate how often or when snapshots are performed
  • **Transactional semantics**: Ensures continuity of jobs through various failures
  • **Scheduling Strategy**: Adapts snapshot frequency to reduce RPO violation

• Future Work
  • Application-consistent snapshots
  • Application consisting of multiple volumes
Thank You!

Umesh Deshpande*, Nick Linck, Sangeetha Seshadri
IBM Research – Almaden, San Jose, CA
Contact*: udeshp@us.ibm.com