



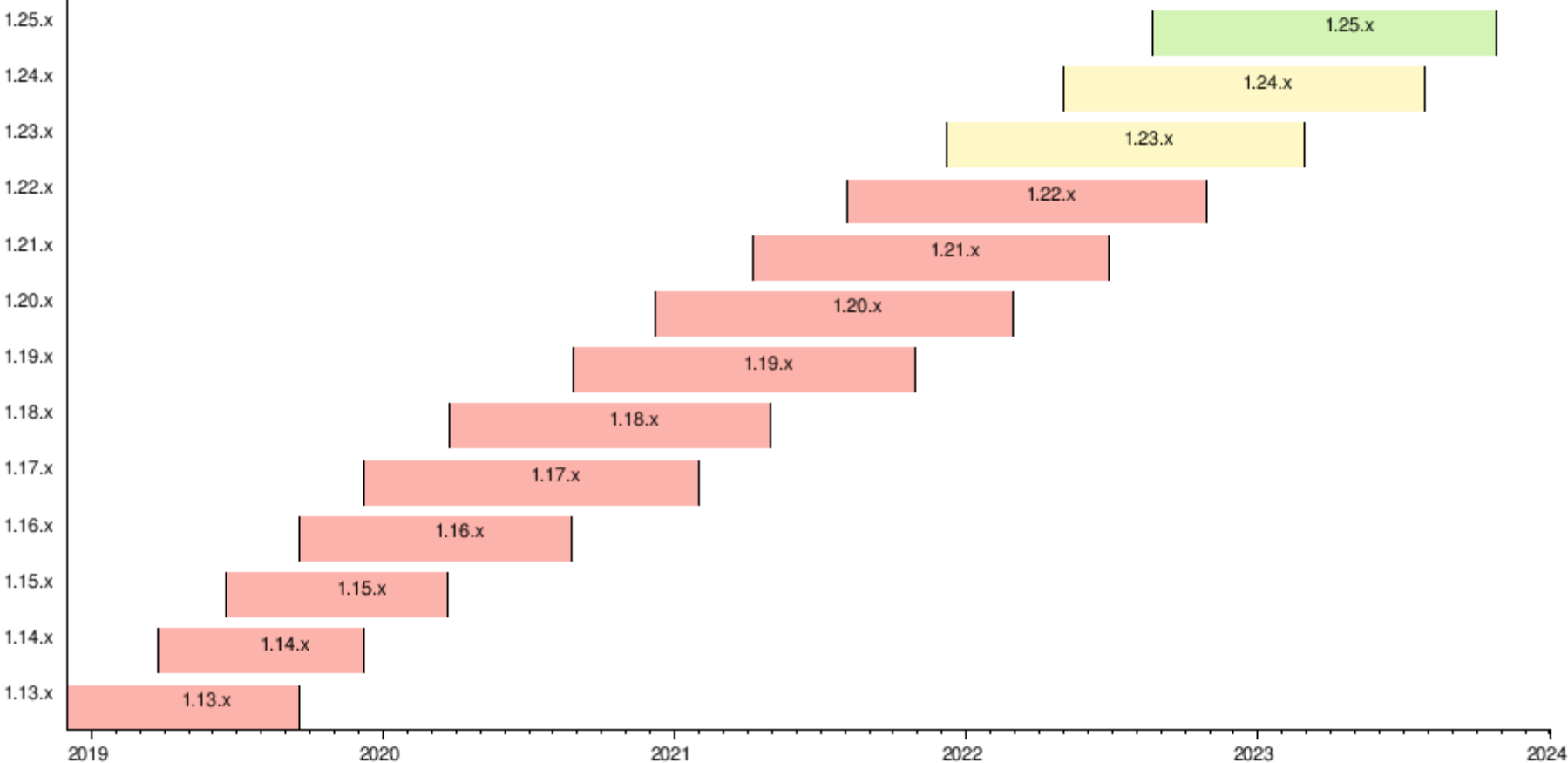
**NetApp**

# Stop worrying about upgrades with smart data management

Fredrik Nygren  
Sr. SE Manager & Field CTO, NetApp Nordics and Baltics



# How scary is this?



**For some of us: Super scary!**

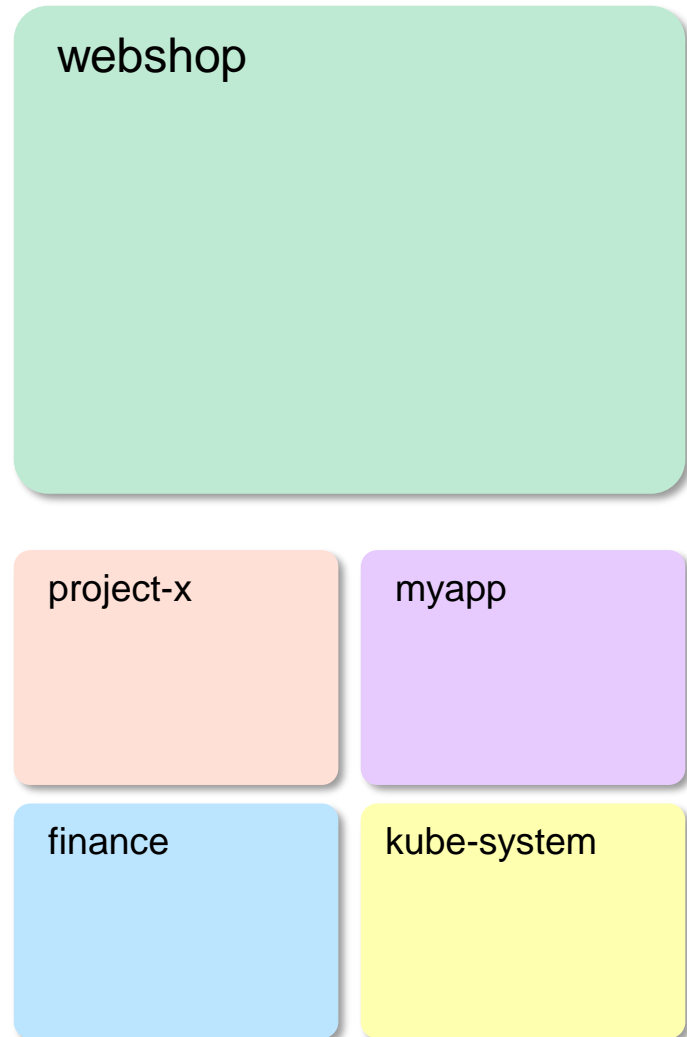
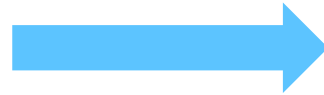


First things first,  
what do we need to  
protect?



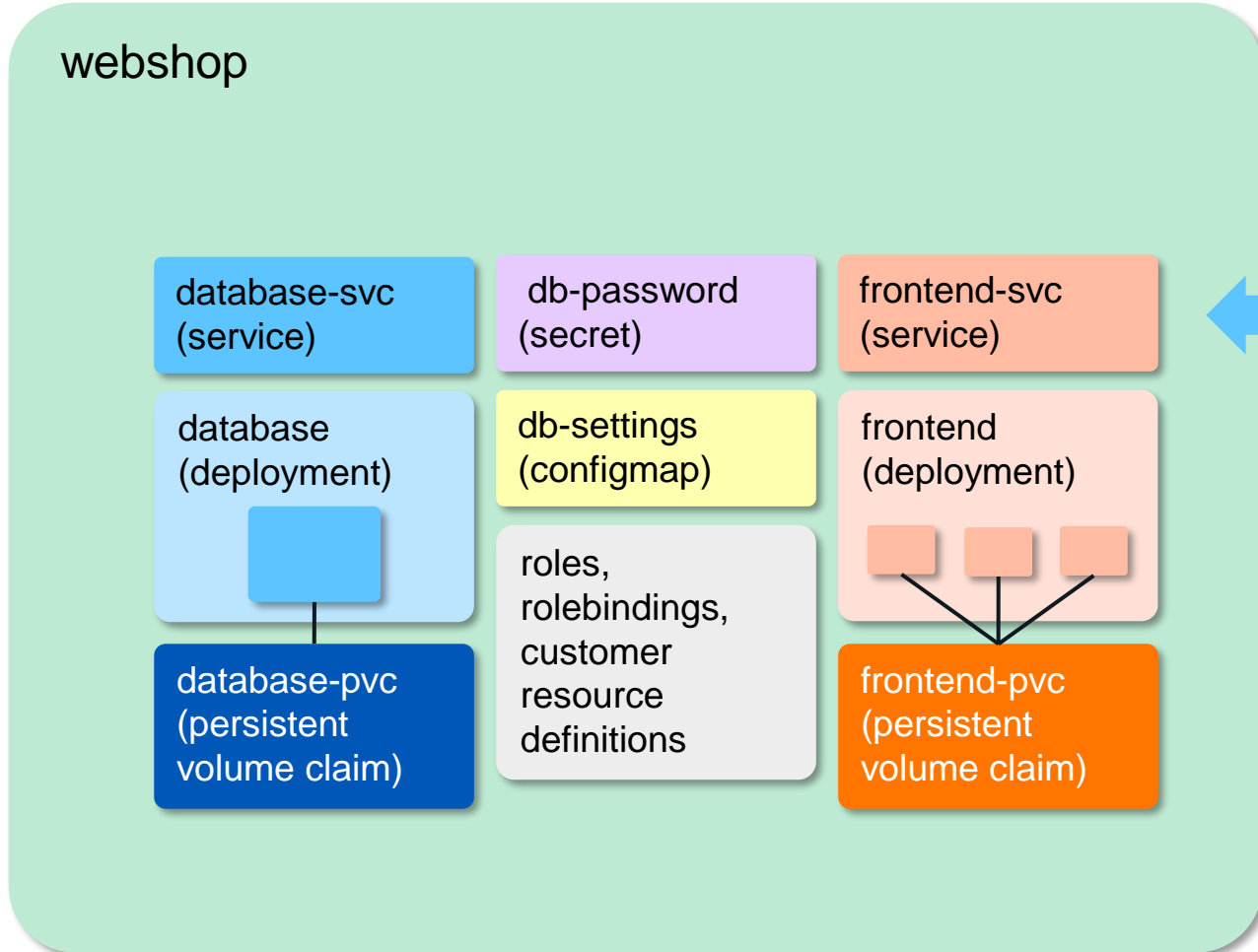
# All resources are organized in namespaces

```
namespace.yaml
---
apiVersion: v1
kind: Namespace
metadata:
  name: webshop
  labels:
    app: webshop
    env: production
```



# A namespace contains resources

All resources are defined using yaml and stored in e.g. Git



```
frontend-svc.yaml
---
apiVersion: v1
kind: Service
metadata:
  name: frontend-svc
  labels:
    app: webshop
    tier: frontend
  namespace: webshop
spec:
  type: LoadBalancer
  ports:
    - name: http
      protocol: TCP
      port: 80
      targetPort: 3000
  selector:
    tier: frontend
```

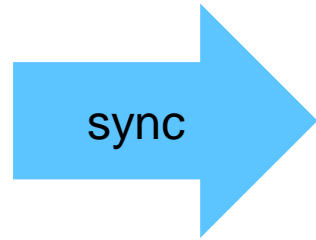


The screenshot shows the Visual Studio Code interface with a file explorer on the left and a code editor in the center. The file explorer shows a project named 'NEO4J-FLASK' with subfolders 'blog', 'static', and 'templates'. The 'k8s' folder is expanded, showing files like '00\_namespace.yaml', '10\_neo4j.yaml', and '20\_flask.yaml'. The code editor displays the content of '20\_flask.yaml', which is a Kubernetes Deployment manifest for a Flask application. The manifest includes fields for apiVersion, kind, metadata (labels and namespace), and spec (replicas, selector, strategy, and template). The terminal at the bottom shows a PowerShell prompt with the command 'git push' entered.

```
k8s > {..} 20_flask.yaml
1  apiVersion: apps/v1
2  kind: Deployment
3  metadata:
4    labels:
5      app: guestbook
6      tier: frontend
7      environment: prod
8    name: flask
9    namespace: guestbook
10 spec:
11   replicas: 1
12   selector:
13     matchLabels:
14       app: guestbook
15       tier: frontend
16       environment: prod
17   strategy:
18     rollingUpdate:
19       maxSurge: 25%
20       maxUnavailable: 25%
21     type: RollingUpdate
22   template:
23     metadata:
24       labels:
25         app: guestbook
26         tier: frontend
27         environment: prod
```

1: powershell  
Windows PowerShell  
Copyright (C) Microsoft Corporation. All rights reserved.  
  
Try the new cross-platform PowerShell <https://aka.ms/pscore6>  
  
PS C:\Users\frny\Code\neo4j-flask> git push





APPLICATION DETAILS

Applications / argocd-dev

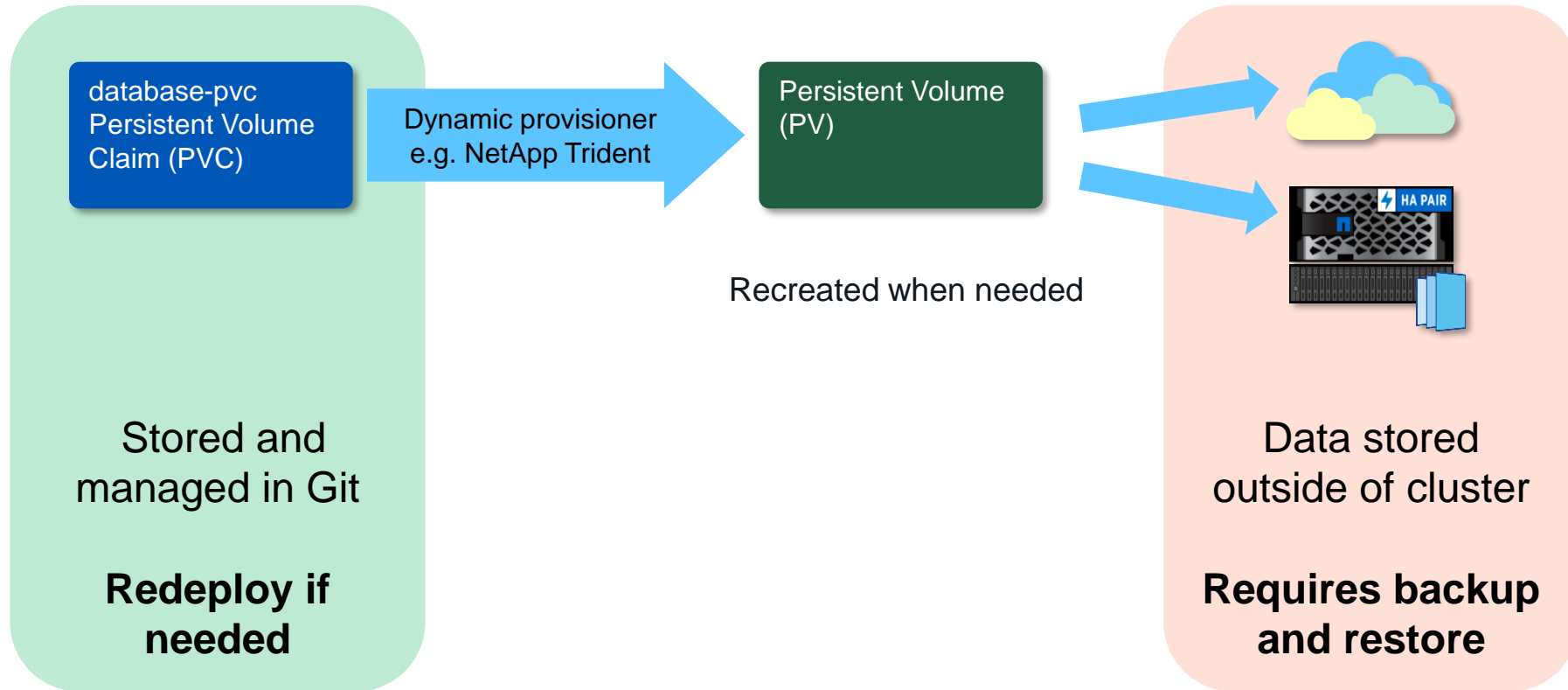
6 DAYS ACTIVE      0 DAYS SINCE LAST SYNCHRONIZED      rollback SUCCEEDED AT 2018-07-17T21:10:55Z

argocd-dev (application)

- application-controller (deployment) → application-controller-5944bbf5fb... (running pod)
- argocd-server (deployment) → argocd-server-566697c64c-jb9vs (running pod)
- argocd-server (service)
- argocd-repo-server (deployment) → argocd-repo-server-5945676c77-4... (running pod)
- argocd-repo-server (service)

## Ok, but what happens to my data?

A PVC is not a volume, it is a developer's request for a volume. Recreating a PVC creates a new request.



Kubernetes has a  
data management  
problem

# Introducing ... Astra



# NetApp Astra

Simplify how you protect, move and store Kubernetes workloads across hybrid and multi-cloud environments

## NetApp Astra



Protect



Move



Store

### Prevent application downtime and data loss

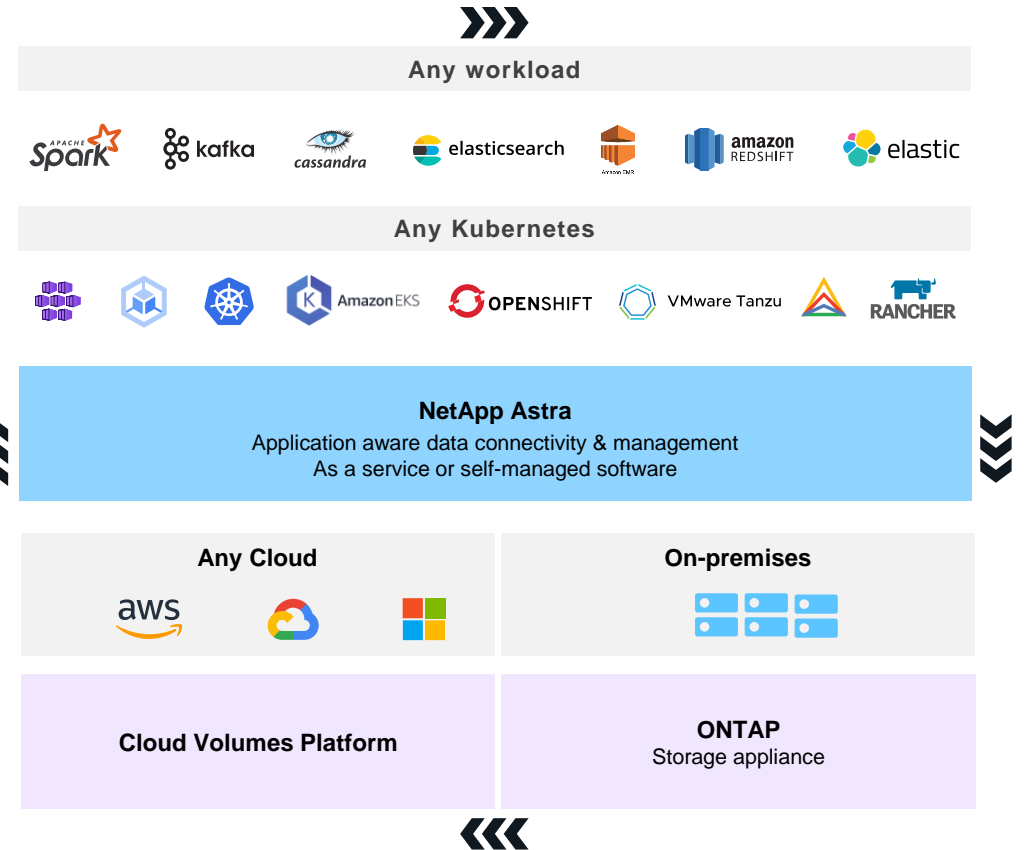
Simple application-aware backups and disaster recovery for your Kubernetes applications

### Reduce toil for time-consuming operations

Effortless cluster upgrades and application portability to simplify dev-test workflows

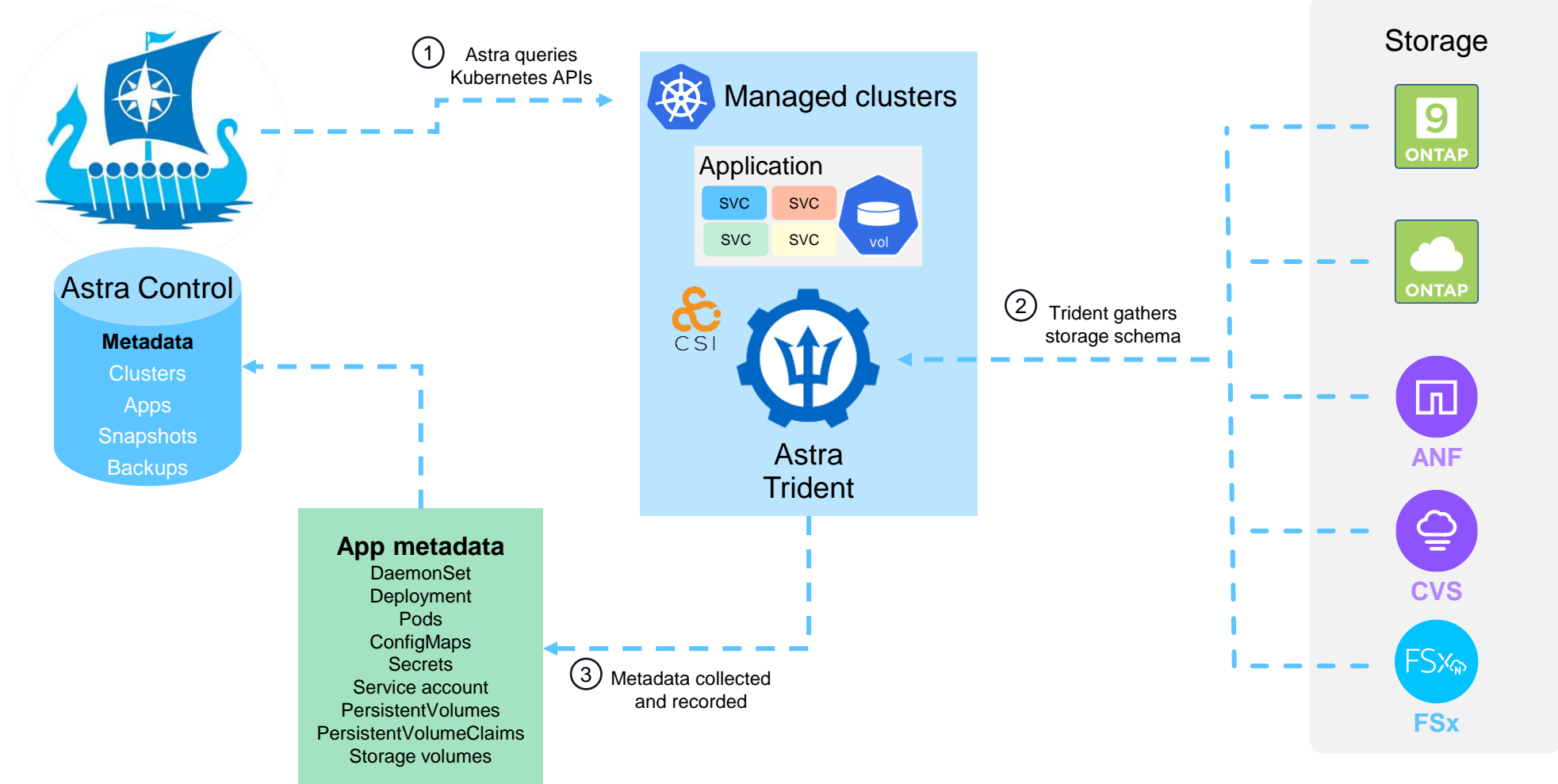
### Scale to fit your needs

Flexible cloud native, shared storage that scales with your applications



# Application Discovery

Astra Control automatically discovers the applications and Kubernetes objects

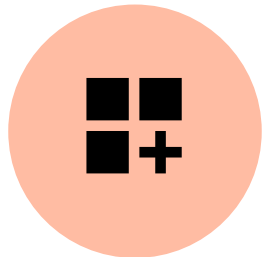


# Managed Applications in Astra Control

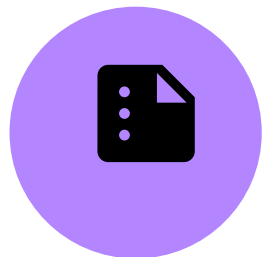
Multiple options to protect applications and data



Manage an application within a namespace or across multiple namespaces



Manage an application based on Kubernetes labels within one or more namespaces

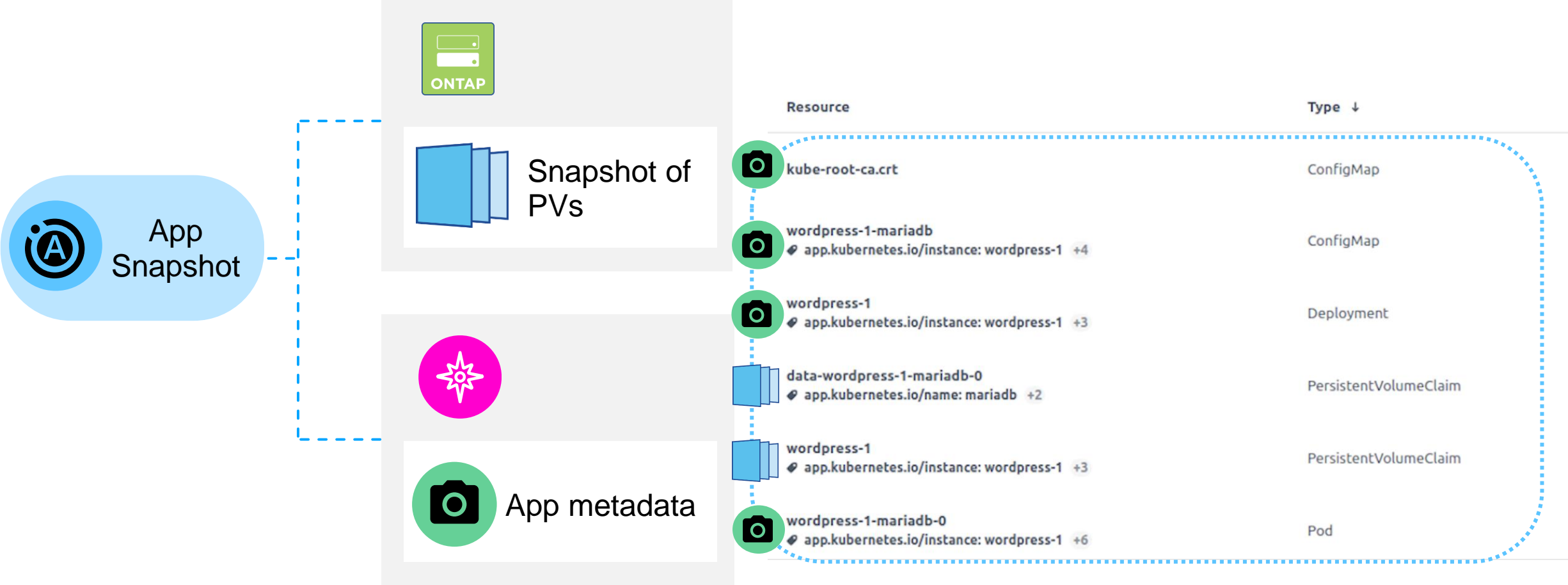


Manage an application along with cluster scoped resources



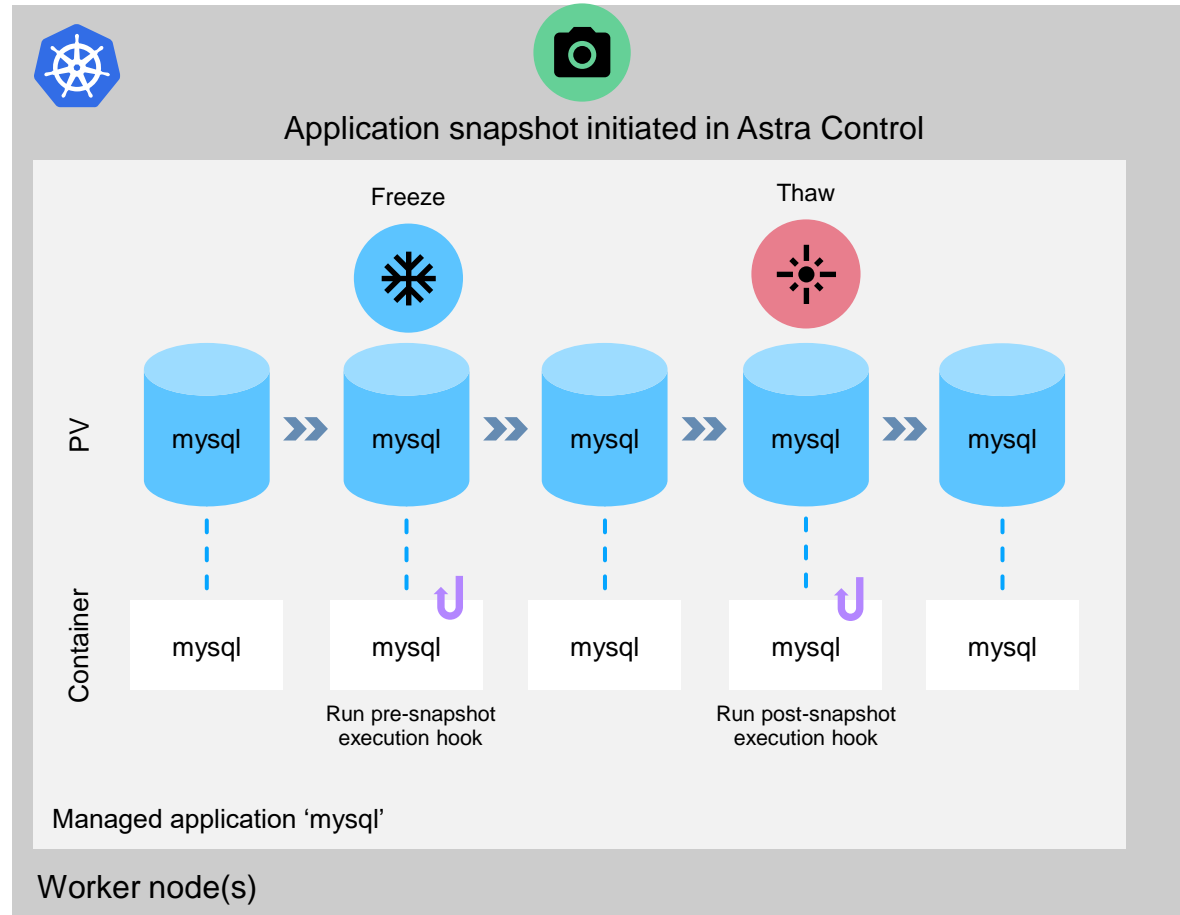
# Data Protection On-Demand or Scheduled

Protect your Application, Metadata and Persistent Volumes



# Custom behavior for snapshots and restores

## Execution Hooks in Astra Control



Application snapshots, backups, and restores can run application specific operations.



Automate application-specific behavior

- Pre-snapshot (freeze) & post-snapshot (thaw)
- Pre and post restore



Application consistent state on disk at time of snapshot and backups.



Application specific steps required to bring application back online during restore.



Library of Execution hooks provided in NetApp sponsored Open-source project Verda.

- MySQL, MariaDB and PostgreSQL, Cassandra
- Kafka, Elasticsearch

# In-place Restore of Application Snapshots and Backups

Entire application restore made easy

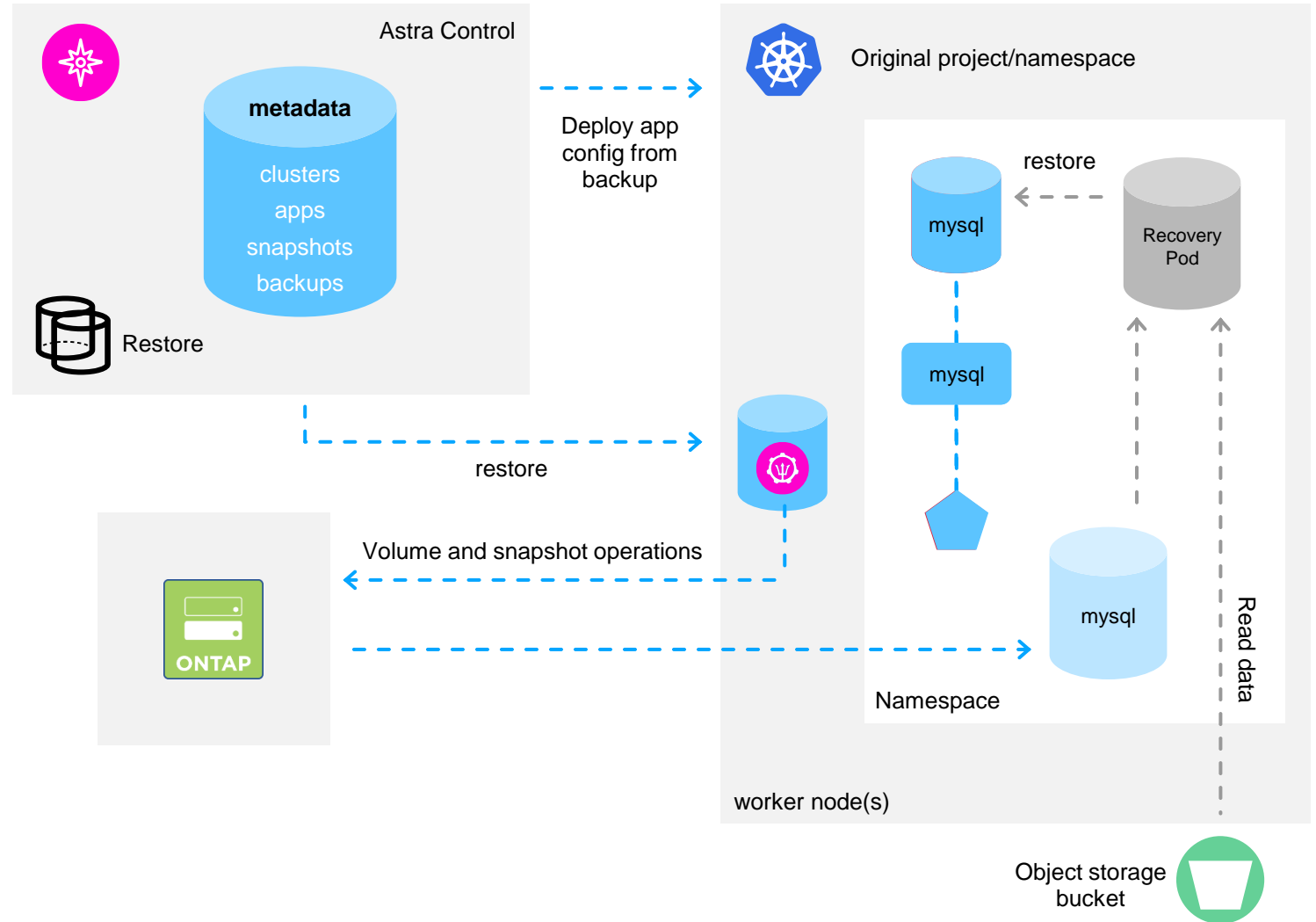
**Restores application to same namespace as original application**

**Works with application snapshots and application backups**

**Replaces existing application with previous state of the application**

Kubernetes objects are recovered

Data on persistent volumes is restored from volume snapshot or volume backup on object storage bucket



# Application mobility



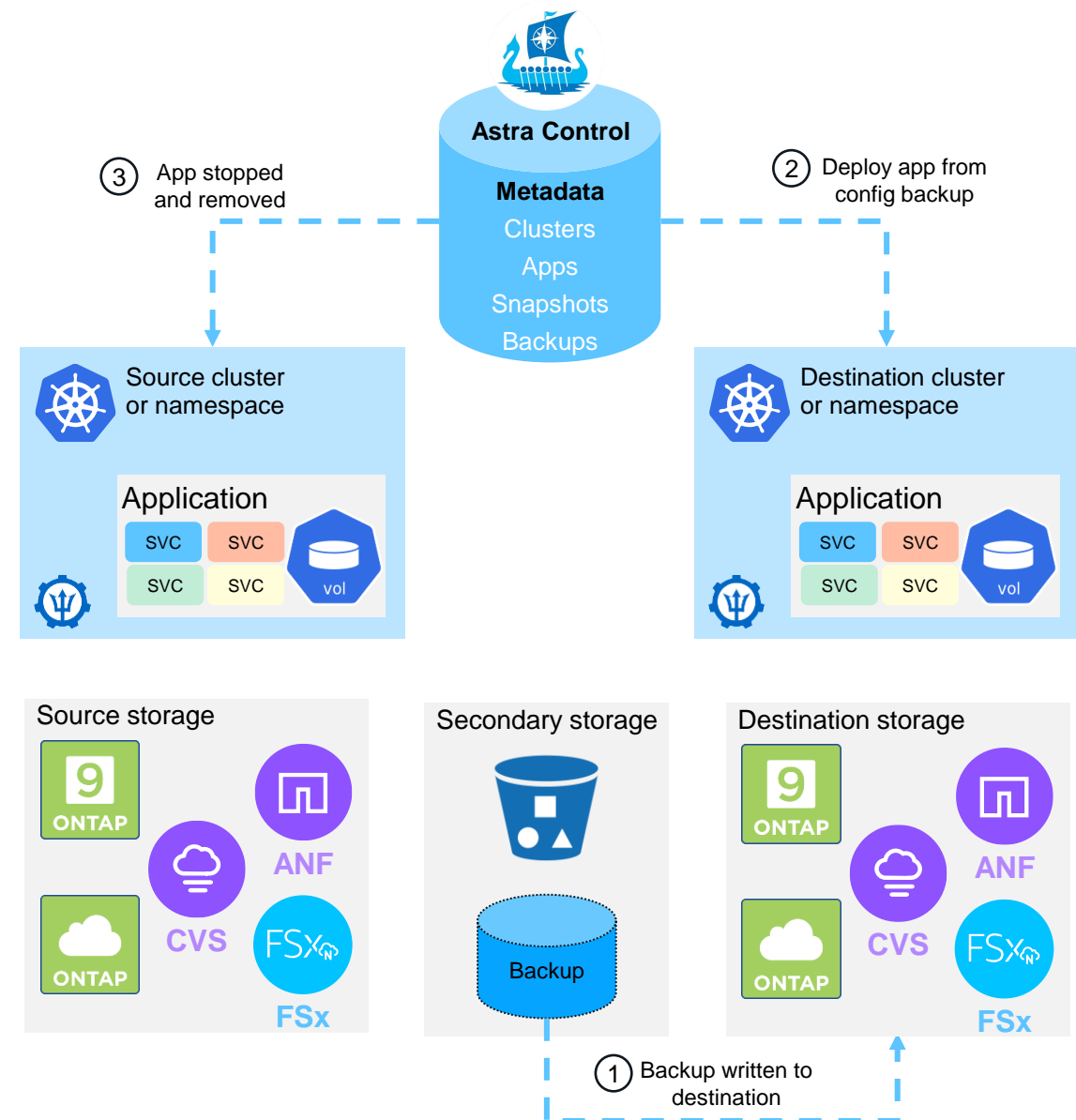
## Problem

Requirement to move data due to data residency, compliance, or regulatory reasons



## Solution

Astra can clone and move application data freeing the app to move between clusters either in the cloud or on-premises



# Key Takeaways

- GitOps is not enough for applications with persistent volumes.
- A lot of software can export it's state, but is it consistent across your application?
- Backups are easy, restores are hard. How quickly can you restore your application to a new cluster?
- NetApp Astra solves all of these problems.