Application Consistent Backup for Containerized Workloads

JACOB JOHN ANDATHETHU (Technical Marketing Engineer [Open EcoSystems])

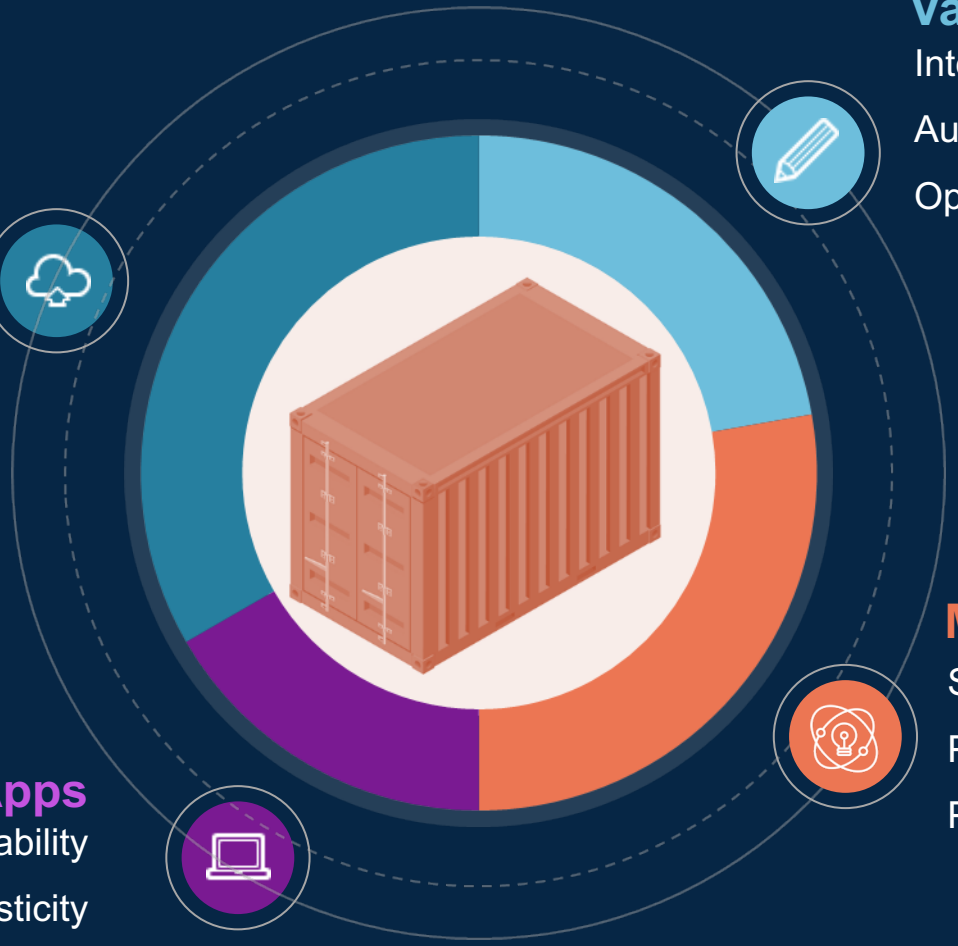
Agenda

1. Stateful containerization of workloads.
2. Existing data protection technologies.
3. Problem statement.
4. Application consistent backup and restore.
5. Disaster Recovery scenario.

Why Containers?

Value Acceleration
Integrate
Automate
Optimize

Cost Savings
Avoid virtualization tax
Efficiency
Productivity



Modernizing Apps
Standardization
Performance
Portability

Cloud-native Apps
Composability
Elasticity
Scalability

Why Storage Persistence?

Containers are ephemeral — data is not

There are no real stateless applications

Data persistence requirements in the enterprise have not changed

12-Factor Methodology [Factor 6]

Factor 6

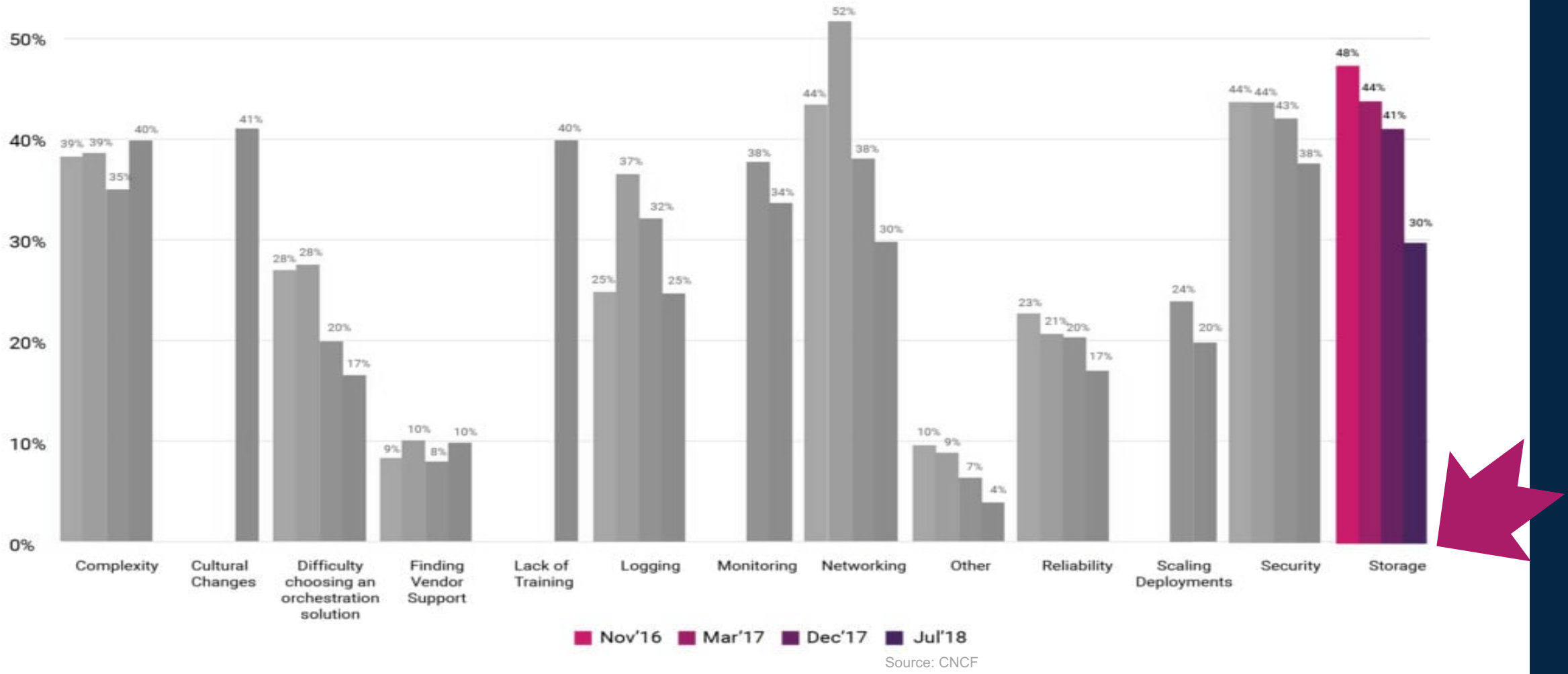
- 12-factor methodology for application development states that applications should be stateless (factor six)
- Data ought to be stored on a backing service.
- Backing service resources like databases requires persistent data.

Containerized Stateful Applications

The top 12 application components running in containers

Key Assessment: The old merges with the new

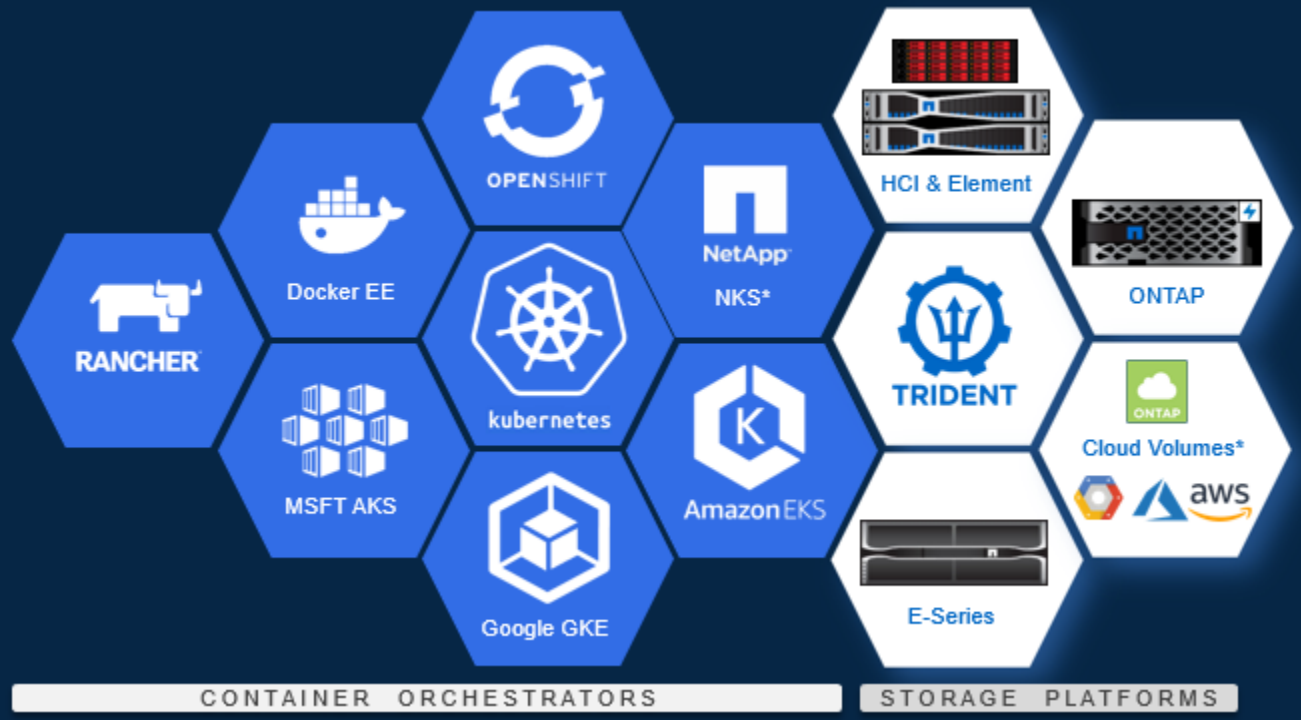
Persistent Storage Remains a Top Challenge for Containers



Source: CNCF

Container Storage Plugin (Dynamic Storage Provisioner.)

TRIDENT



Open Source
Maintained and supported by NetApp

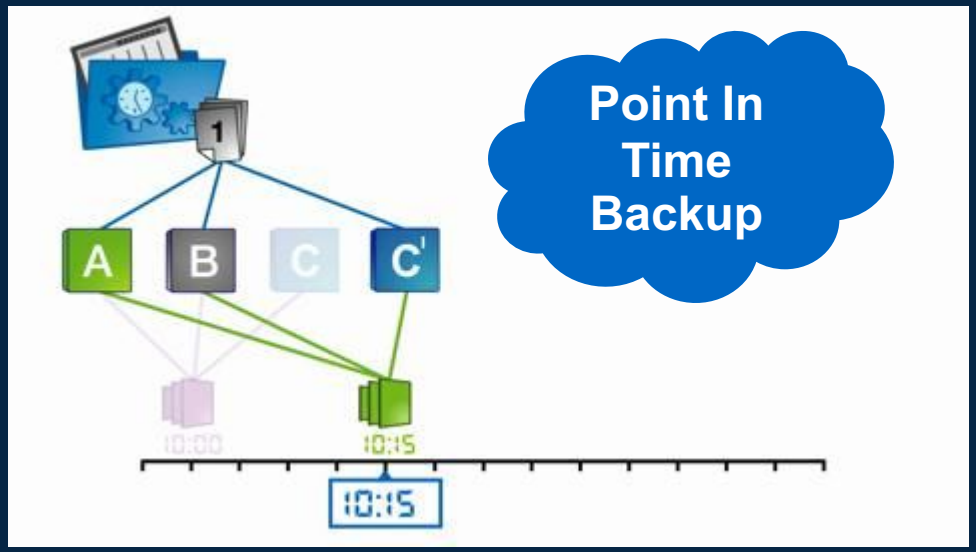
Storage orchestrator
Simplifies storage consumption by end users through Kubernetes

Cross-Platform
Supports all major NetApp storage products and services

Data Protection Solutions for Containerized Workload

Solution 1:- Storage Container Plugin Snapshots (Crash Consistent Backups)

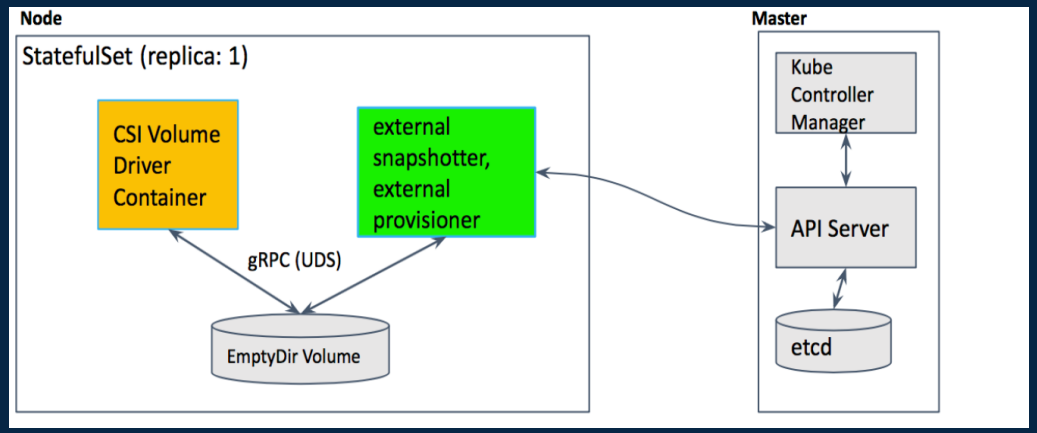
- Storage hardware vendors have Storage Container Plugin for dynamically provisioning volumes from the hardware.
- Snapshotting capabilities are in-corporated into the Docker or Kubernetes storage container plugins.
- Volume Snapshots are taken by assigning volumes to a specific snapshot policy .
- Snapshots policy can be configured to a volume while creating the PVC.



Data Protection Solutions for Containerized Workload

Solution 2:- Storage Container Plugin CSI Standard (Crash Consistent Backups)

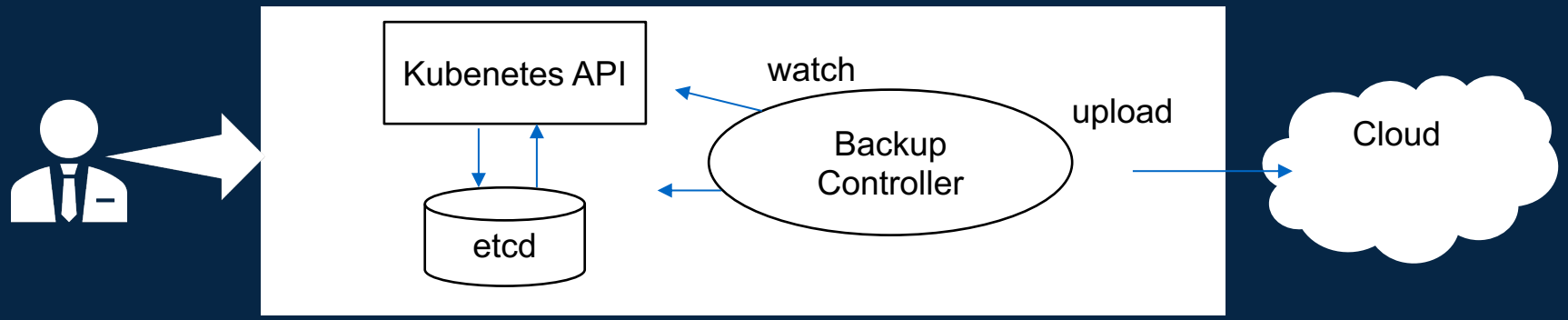
- Storage hardware vendors have Storage Container Plugin that conforms to the Container Storage Interface Spec(CSI)
- Volume snapshotting support was introduced in Kubernetes v1.12 as an alpha feature.
- Similar to **PersistentVolume** and **PersistentVolumeClaim** APIs which are used to provision volumes, **VolumeSnapshotContent** and **VolumeSnapshot** API resources are provided to create volume snapshots.
- This provisioning is based on **VolumeSnapshotClasses**



Data Protection Solutions for Containerized Workload

Solution 3:- Open Source Tools (Crash Consistent Backups)

- Helps in taking backup and pushing the data to the cloud provider of choice.
- Backup controller watches the backup request from the users.
- BackupController begins the backup process and collects the data to backup by querying the API server for resources.
- BackupController makes a call to the object storage service to upload the backup file.



Data Protection Solutions for Containerized Workload

Solution 4:- Third Party Solutions (Crash Consistent Backups)

- Backup Software Tools focuses mainly backup of containers and images.
- Streaming backups of docker containers and images found on a specific clients.
- Backup creates a tar archive for the image and saves it to the MediaAgent.
- Restores are done by uploading the image.tar file directly from the MediaAgent to the Docker host.

Problem Statement.

- Crash Consistent Backup/Restore
  - Backup doesn't not capture data in memory or any pending I/O operations
  - Restoring requires extra work, such as journaling forward, before an application can be brought back online.
- How do we do an Application Consistent Backup/Restore for containerized workload?

How do we take an Application Consistent Backup?

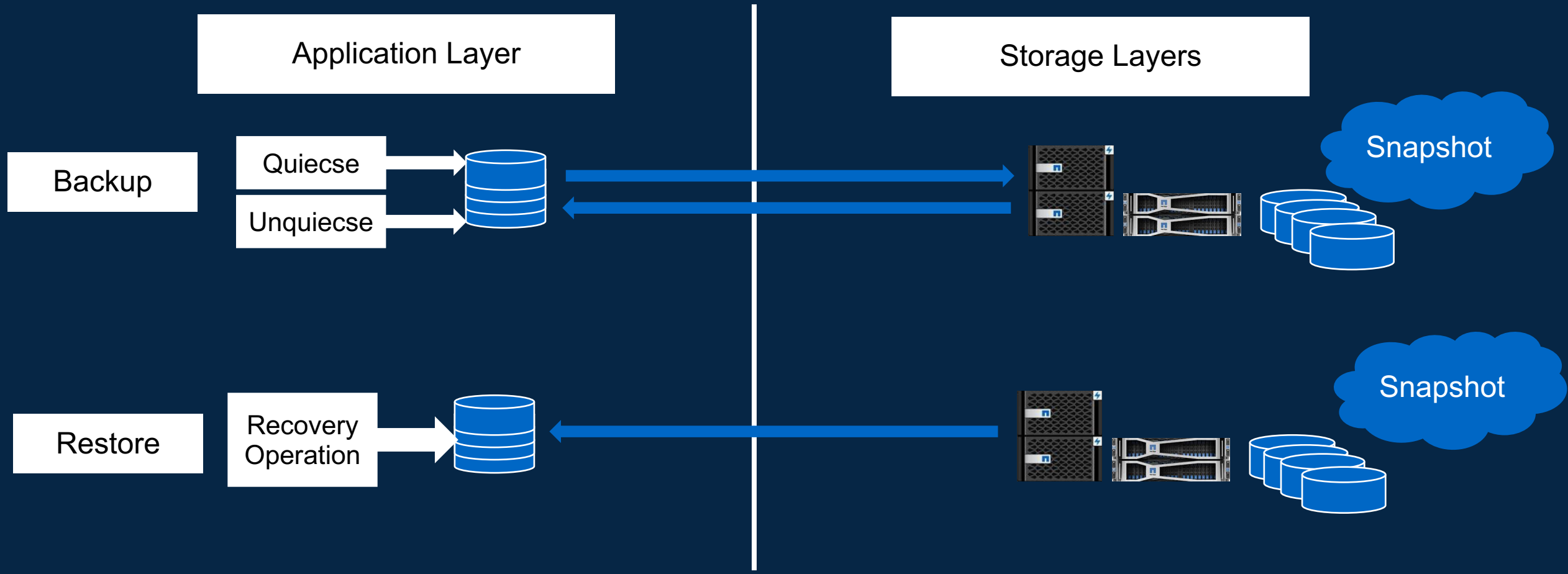
How do we Restore and Recover Applications?

How does Disaster Recovery for Applications work?

Application Consistent Backup,Restore and DR.

How does Application Consistent Backup and Restore work?

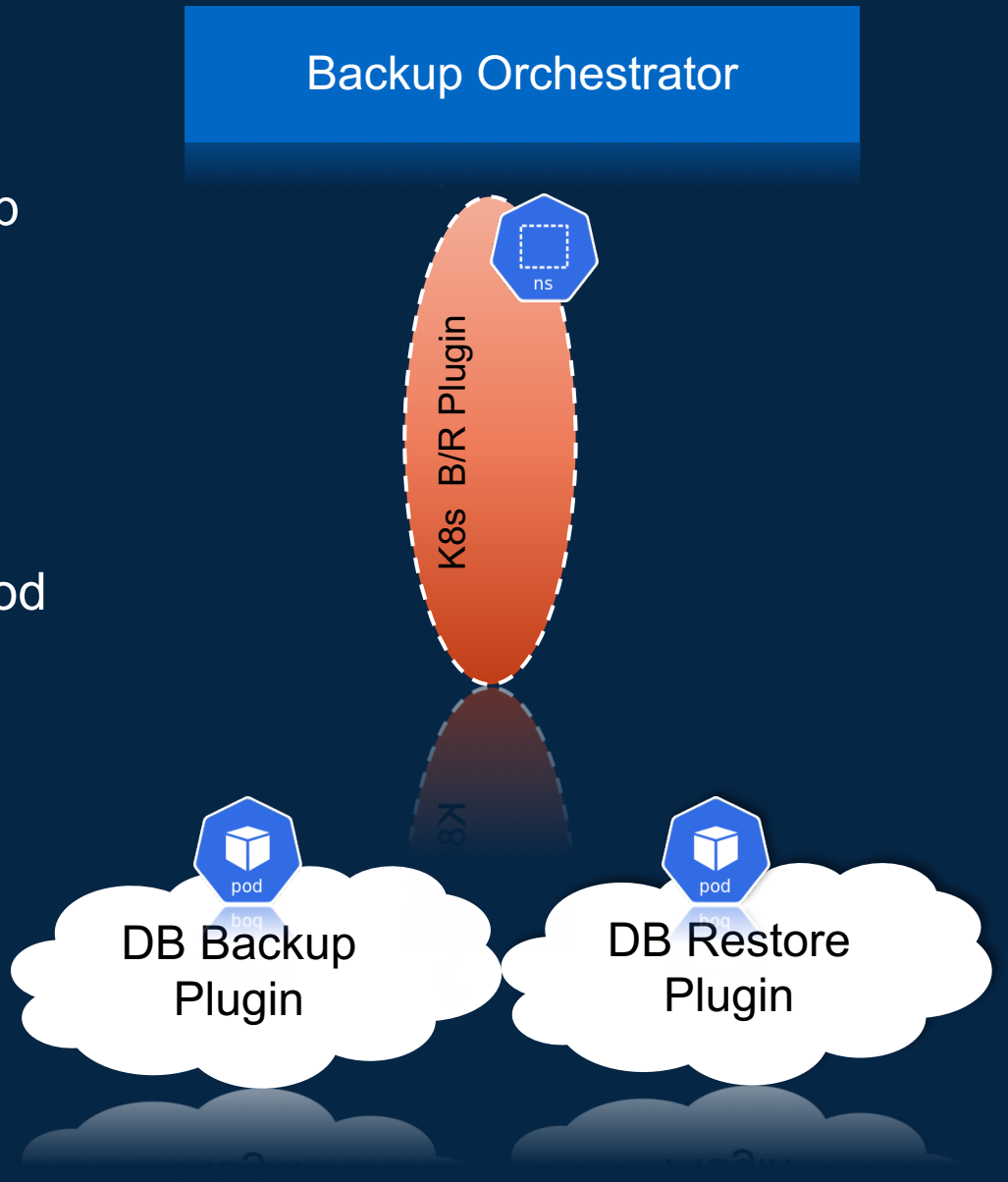
- App consistent backup and restore operations involve two different layers.



Application Consistent Backup,Restore and DR.

Components

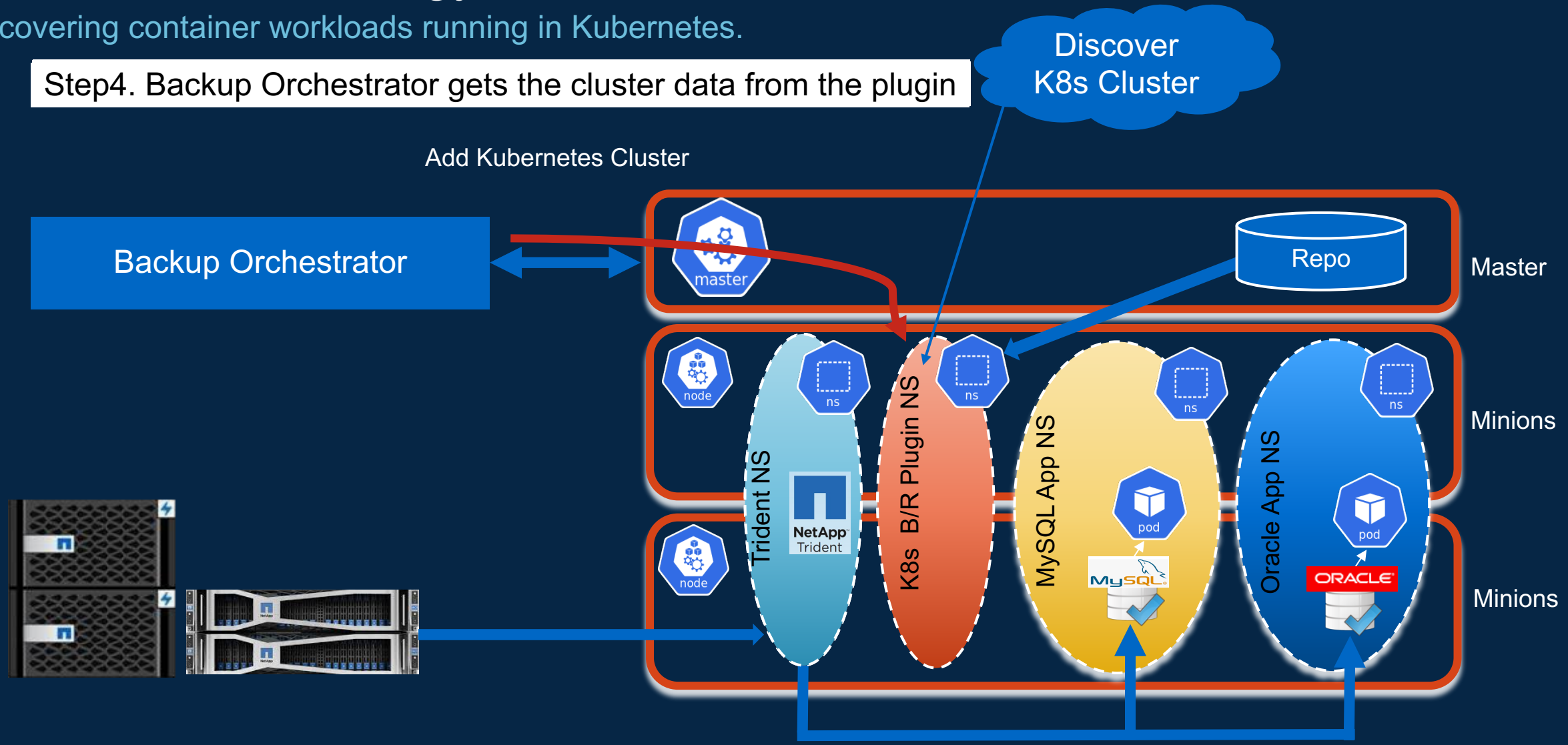
- Workflow makes use of Backup Orchestrator (Takes snapshots on the storage, metadata handling, setting up K8s B/R plugin.)
- The K8s backup plugin is run as deployment in its own namespace.
- The DB Backup and Restore plugin is deployed as a pod in the required namespace.
- The DB Backup plugin takes care of quiescing and unquiescing.
- The DB Restore plugin takes care of running recovery operations.



Discover Methodology

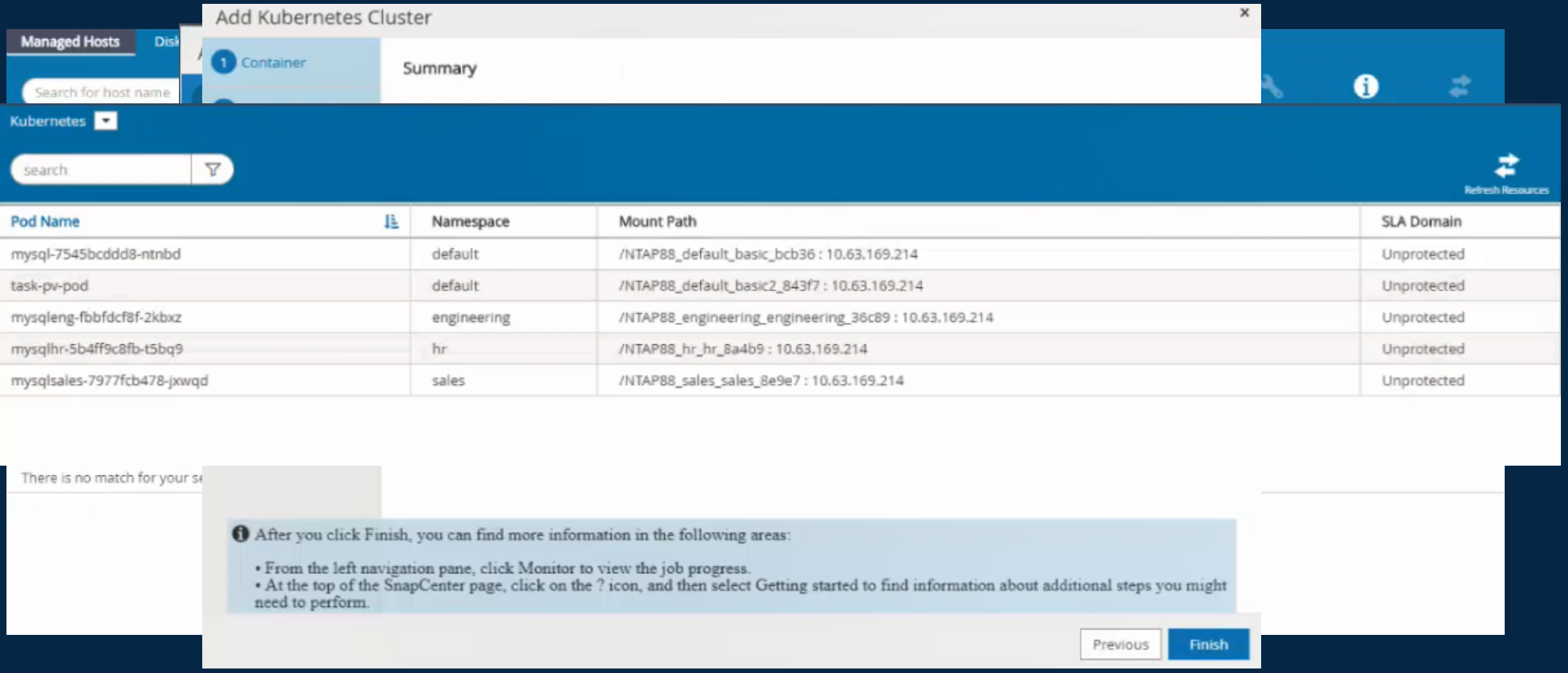
Discovering container workloads running in Kubernetes.

Step4. Backup Orchestrator gets the cluster data from the plugin



Discover Methodology

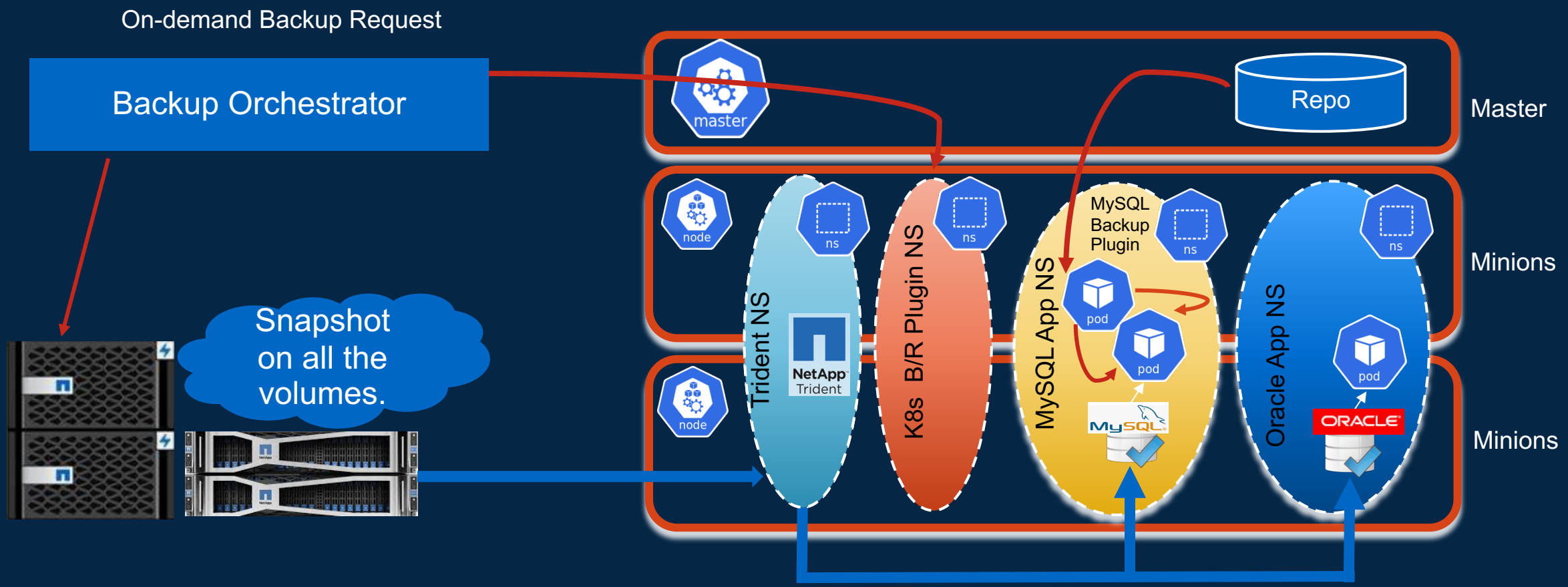
Discovering container workloads running in Kubernetes.



Backup Methodology

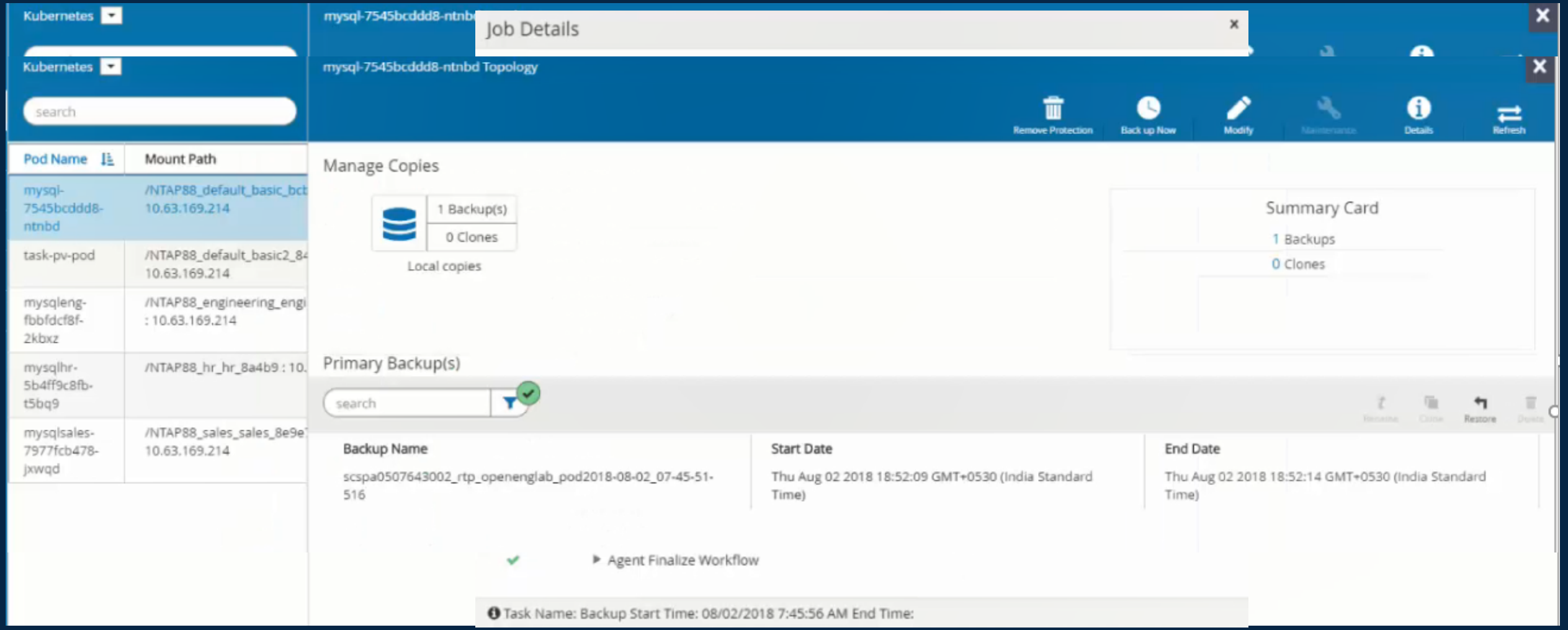
Application Consistent backup for Containerized Workload

Step7. Remove the backup plugin.



Backup Methodology

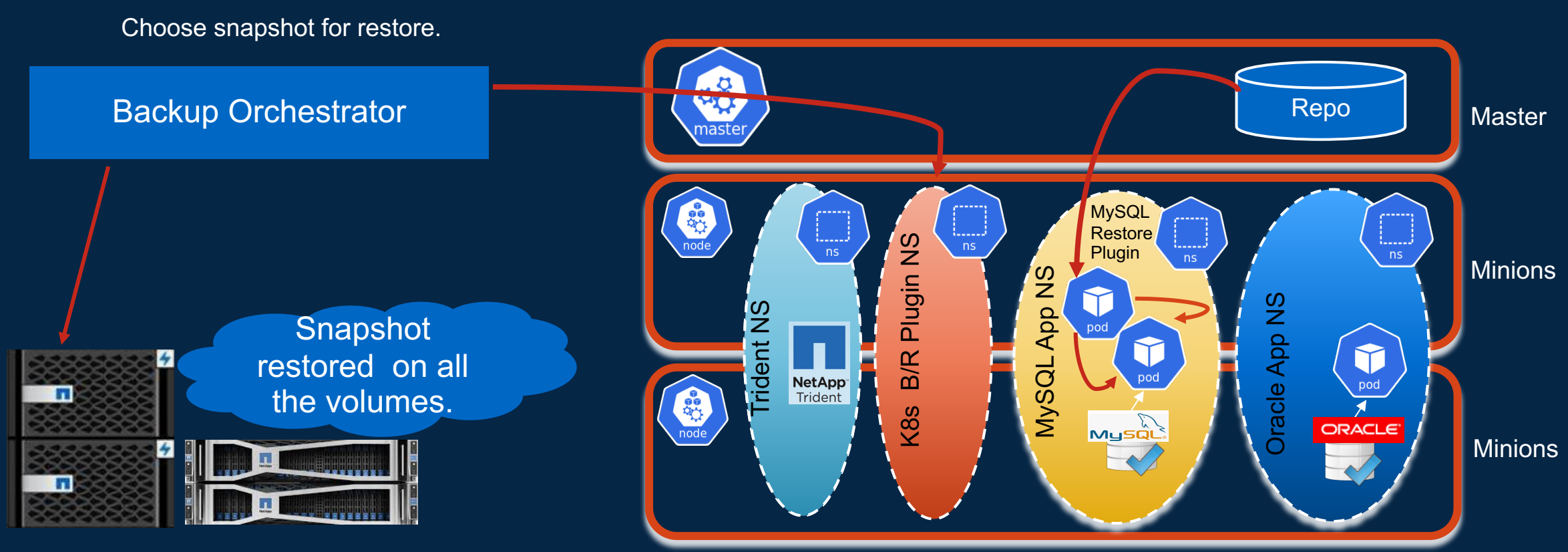
Application Consistent backup for Containerized Workload



Restore Methodology

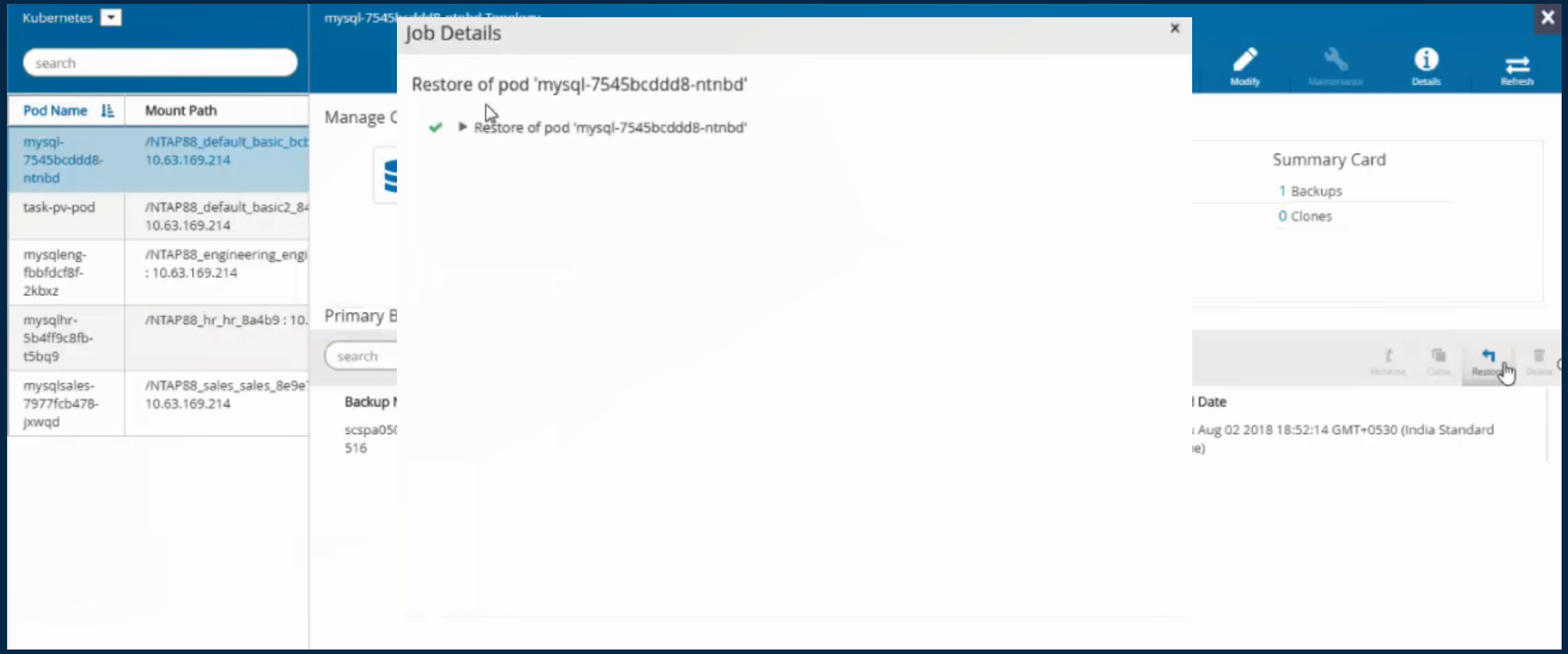
Application Consistent Restore for Containerized Workload

Step8. The restore plugin will be cleaned up.



Restore Methodology

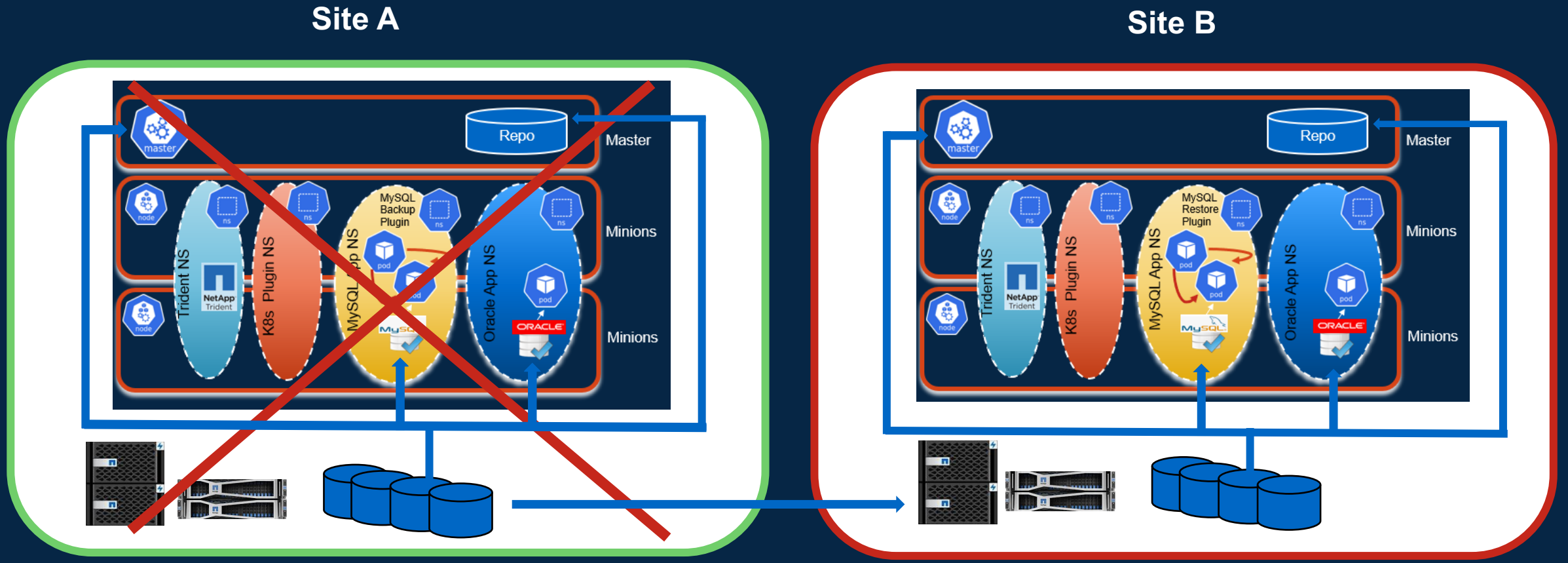
Application Consistent Restore for Containerized Workload



Disaster Recovery Scenario

Disaster recovery for Containerized Applications.

Step4 Kubernetes and Applications are deployed on K8s with the Persistent volume behind.



Summing Up

- Existing backup technologies for containerized workflows.
- What is application consistent backup…
- How to do a application consistent backup and restore…
- A quick look into the disaster recovery scenario.

thePub

NetApp's Developer and Open Source Community

Engage NetApp Engineers, Partners, and Customers

thePub
netapp.io

Slack
netapp.io/slack

Twitter
@NetAppPub

NetApp
nt-ap.com/2CneVeR

GitHub
github.com/NetApp/trident