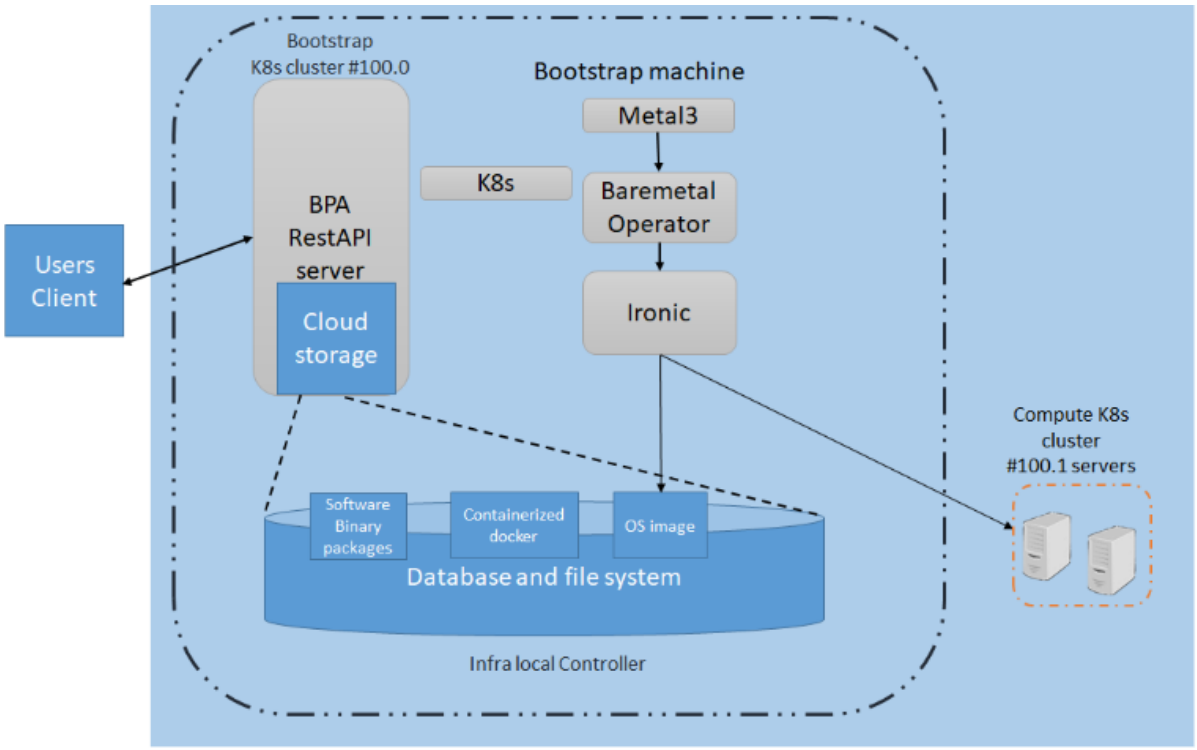
Cloud Storage

Storage Service for Local Controller

The requirement initial from RESTful API Binary Provisioning Agent which provide interface to operator to upload image objects.

There are 3 key functionality for RESTful API Agent:

1. Image provisioning (upload)
2. REST API (POST/PUT/PATCH/DELETE etc)
3. Support resumable upload

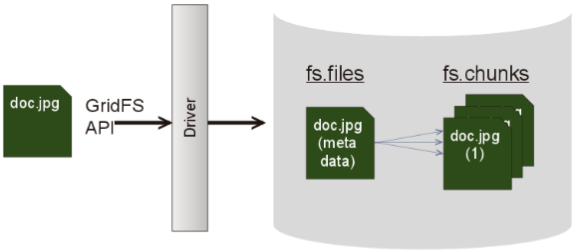


Backend Storage Solutions

There are several considerations for the backend storage solution:

1. Data reliability: some mechanism like replication to make sure the data can be recovered.
2. Cloud native: Support docker or Kubernetes deployment.
3. Amazon S3 API compatible: Support S3 API which is factual standard.
4. Multipart upload is better: which can speed up or resume when upload thread is break.

Solution 1 - GridFS



1. Conception
   a. A simple file system abstraction on top of M
   b. For storing files larger than 16 MB
2. Where to use
   a. If your filesystem limits the number of files in a directory
   b. When you want to keep your files and metadata automatically synced and deployed across a number of system and facilities
   c. When you want to access information from portions of large files without having to load whole files into memory
3. Where don't use
   a. If you need to update the content of entire file atomically
   b. File smaller than 16 MB BSON document size limit, use BinData type to store.
   c. Hard to extend to cloud storage, complex in config and construct for supporting

Solution 2 - MinIO

1. Conception
   a. High performance, open source, Amazon S3 compatible Object Storage server
   b. Designed for AI and big data
2. Features
   a. Well support for cloud storage solution, cloud native design
   b. Client API libraries available for common languages
   c. Security access policy, encryption
   d. Multipart upload and checkpoint download
   e. Server mode and gateway mode
3. Where to use
   a. Best suited for storing unstructured data such as photos, videos, log files, backups and container/VM images
   b. Size of an object can range from a few KBs to a maximum of 5TB.

Implementation

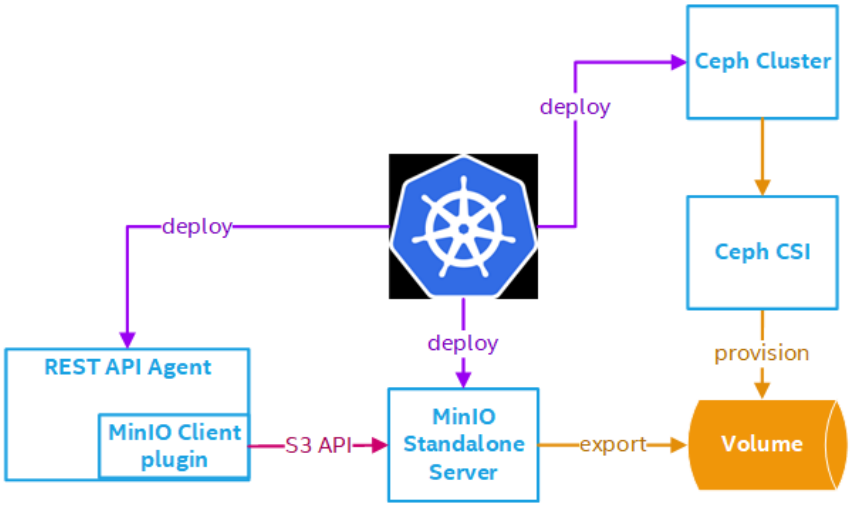
We choose **MinIO** as the backend storage solution which provide Cloud Storage Service for Local Controller.

The implementation for the module is to develop MinIO client plugin based on MinIO Golang API, and integrate with REST API agent, include interface for MinIO client plugin as:

1. Initialize(), which create new client and setup buckets
2. PutImage(), put the object file to MinIO server, support /PUT
3. PatchImage(), Put the object with offset and compose, support /PATCH
4. DeleteImage(), Delete the image
5. CleanupImages(), Cleanup and remove incomplete upload threads

Deployment

the reliable volume is provided by Ceph cluster with CSI, and which used by MinIO standalone server, MinIO clinet plugin code interact with S3 API to provide object service for REST API agent.

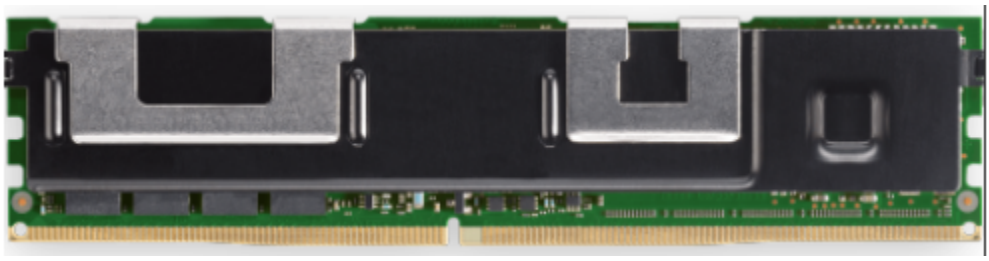


Optane PM (Persistent Memory) Plugin

Optane PM and PMEM-CSI

The term persistent memory is used to describe technologies which allow programs to access data as memory, directly byte-addressable, while the contents are non-volatile, preserved across power cycles. It has aspects that are like memory, and aspects that are like storage.

Intel Optane DC Persistent Memory is an innovative technology that delivers a unique combination of affordable large memory capacity and persistence (non-volatility). The persistent memory technology can help boost the performance of data-intensive applications, such as in-memory analytics, databases, content delivery networks, and high performance computing (HPC), as well as deliver consistent service levels at scale with higher virtual machine and container density.



To enable the Optane PM (Persistent Memory) on cloud native application, Intel has develop the PMEM-CSI driver for Kubernetes. Intel PMEM-CSI is a CSI (Container Storage Interface) storage driver for container orchestrators like Kubernetes. It makes local persistent memory (PMEM) available as a filesystem volume to container applications. It can currently utilize non-volatile memory devices that can be controlled via the libndctl utility library.

The PMEM-CSI driver can operate in two different device modes: LVM and direct.

1. In Logical Volume Management (LVM) mode the PMEM-CSI driver uses LVM for logical volume Management to avoid the risk of fragmentation. The LVM logical volumes are served to satisfy API requests. There is one volume group created per region, ensuring the region-affinity of served volumes.
2. In direct device mode PMEM-CSI driver allocates namespaces directly from the storage device. This creates device space fragmentation risk, but reduces complexity and run-time overhead by avoiding additional device mapping layer. Direct mode also ensures the region-affinity of served volumes, because provisioned volume can belong to one region only.

For the Optane PM plugin in ICN, we support LVM mode, which provide CSI plugin and CSI driver, the framework as following chart.

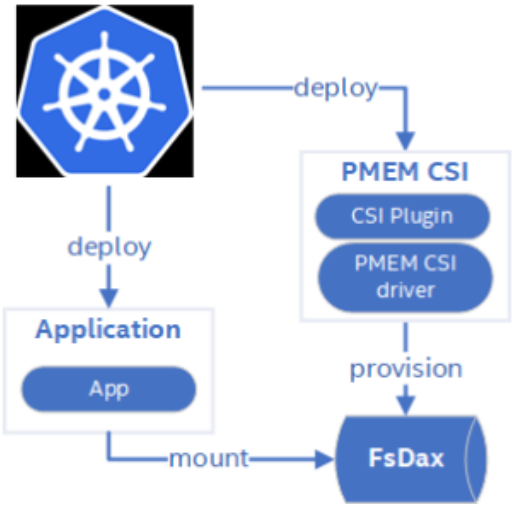
blocked URL

The common usage scenario for Optane PM is mount as cache partition or store metadata for filesystem or clusters (Ceph for example), this work will be design in future release.

Implementation

Optane PM plugin is part of KUD plugin addons, the bring up process defined in KUD deployment_infra ansible playbook.

Optane PM CSI plugin and driver layout as following chart, we create example application with StorageClass and PVC for dynamic volume provisioning.


deploy
PMEM CSI
CSI Plugin
PMEM CSI driver
deploy
Application
App
provision
mount
FsDax