STORAGE DEVELOPER CONFERENCE
SDC 22
Fremont, CA
September 12-15, 2022
BY Developers FOR Developers
A SNIA Event

Building an Object Based STaaS Solution with Poseidon Storage

Swati Chawdhary
Abdul Ahad Amir
Sandeep Agarwal
Jun-HO Jang

Agenda

- Poseidon Introduction
- Object based STaaS
- How to build STaaS with Kubernetes
- STaaS Demo
- Poseidon STaaS performance benchmark
- Conclusion

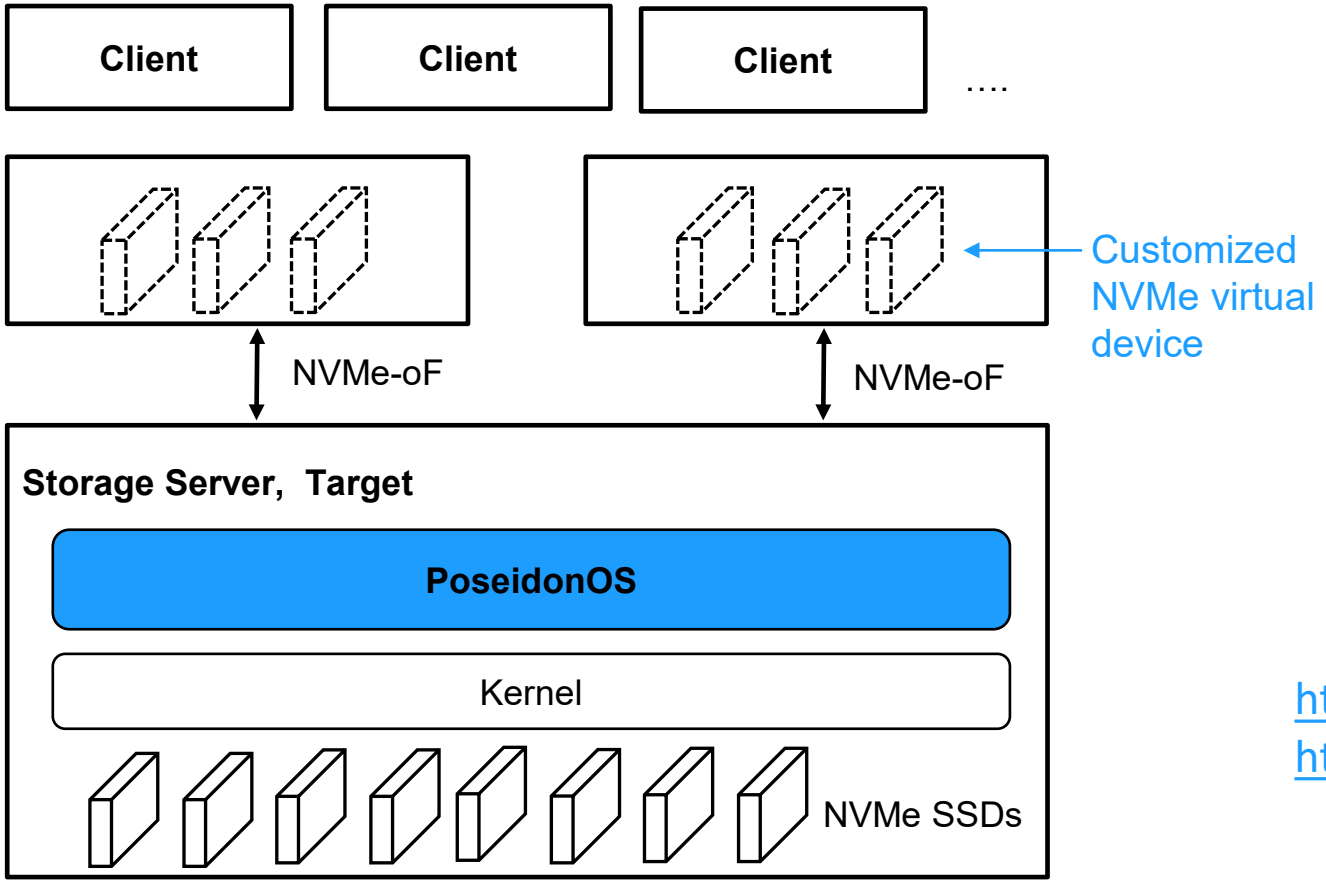
Poseidon Overview

Poseidon Project

- OCP based industrial collaboration b/w “component vendor-System vendor-Data Center targeted for Cloud and Hyperscale data centers
- Open-source H/W & S/W project to expand NVMe eco-system
- It supports a composable architecture based upon the U.2/E1.S/E3.S SSD form factor for enabling storage disaggregation using NVMe Fabrics protocol.

Poseidon OS

- Provide ‘customized’ virtual devices to initiators via NVMe-oF interface



※ Example of customized options for each virtual SSDs
- Capacity
- Performance (IOPS, BW, QoS)
- Features
  - RAID (1, 5, 6, …)
  - Compression
  - Thin Provisioning
- … and MORE!

https://poseidonos.io/
https://github.com/poseidonos/poseidonos

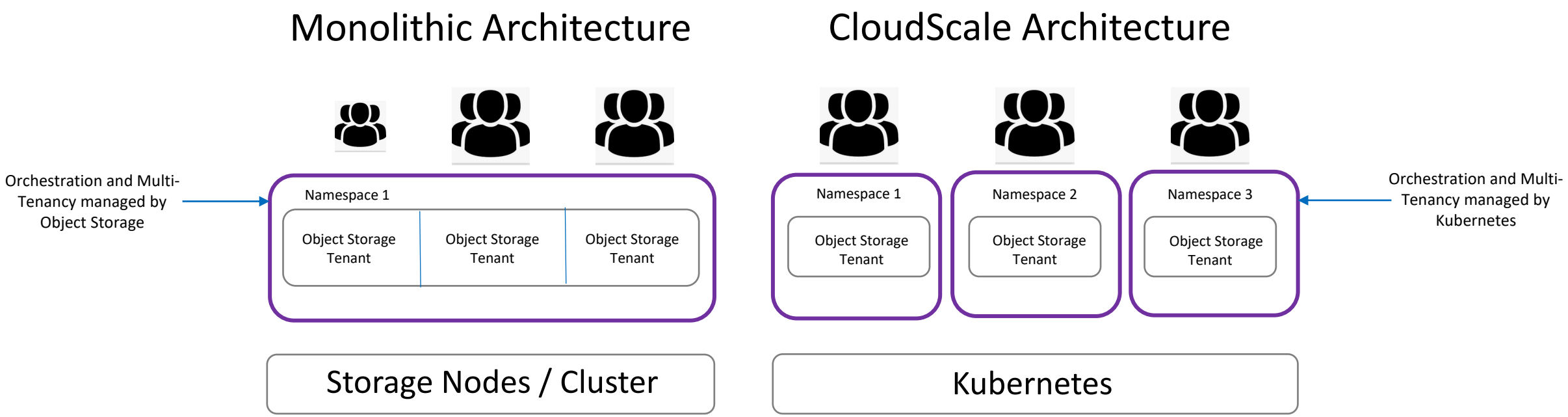
Goal of Poseidon OS

- High Performance
  - Provide high performance volume to each client
- QoS
  - Provide stable performance to each client
- High Availability
  - Guarantee data from hardware and software errors (ex. RAID, 2-node HA)
- Maintenance
  - Provide various features (CLI CMD, RESTful API, …)

Object based STaaS

- STaaS(Storage as a Service) is a data service provided by Cloud Service Providers(CSPs) to allow users to rent storage resources on need basis.
- Object based STaaS service is becoming popular, as object storage is today the dominant class of storage for the cloud
- CSPs are adopting object storage as their primary storage service
- Our goal is to build an object based STaaS platform, optimized for Poseidon storage.

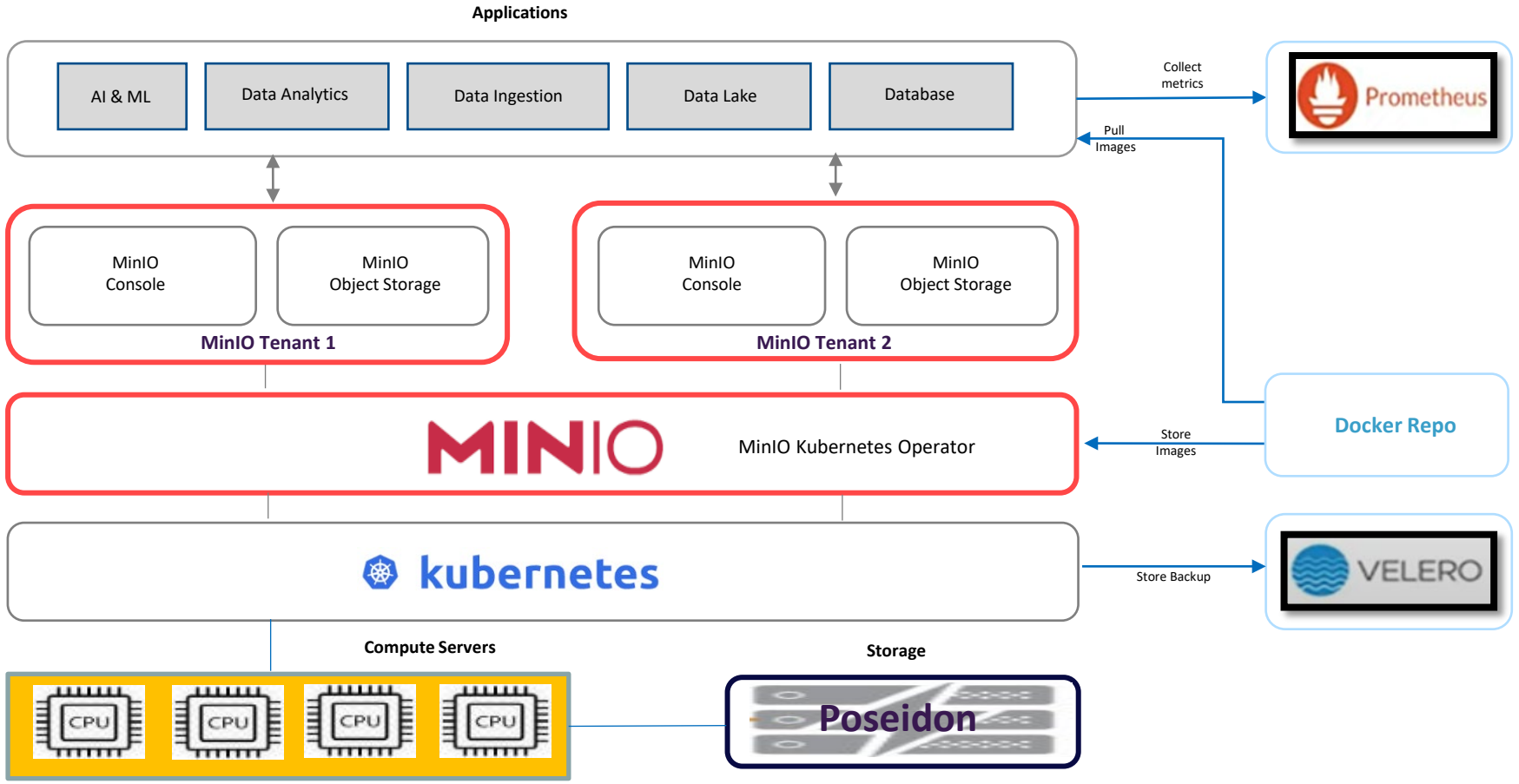
Monolithic vs Cloud Scale deployment



Monolithic vs Cloud Scale deployment

MinIO Object based STaaS Solution on Poseidon

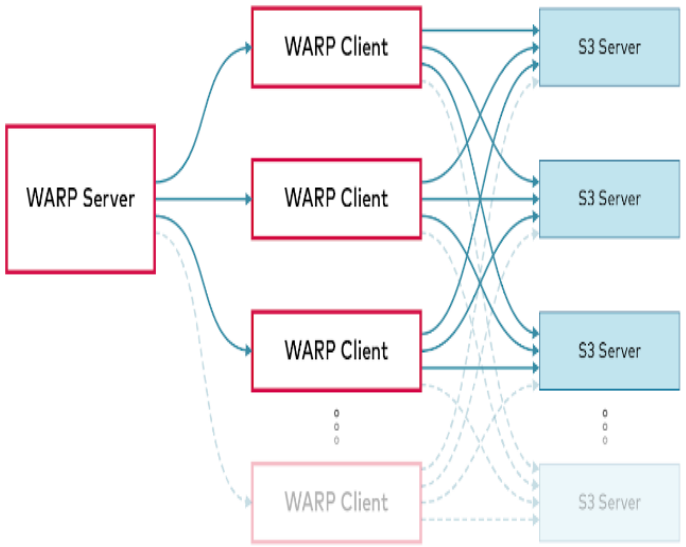
MinIO Object based STaaS Solution on Poseidon



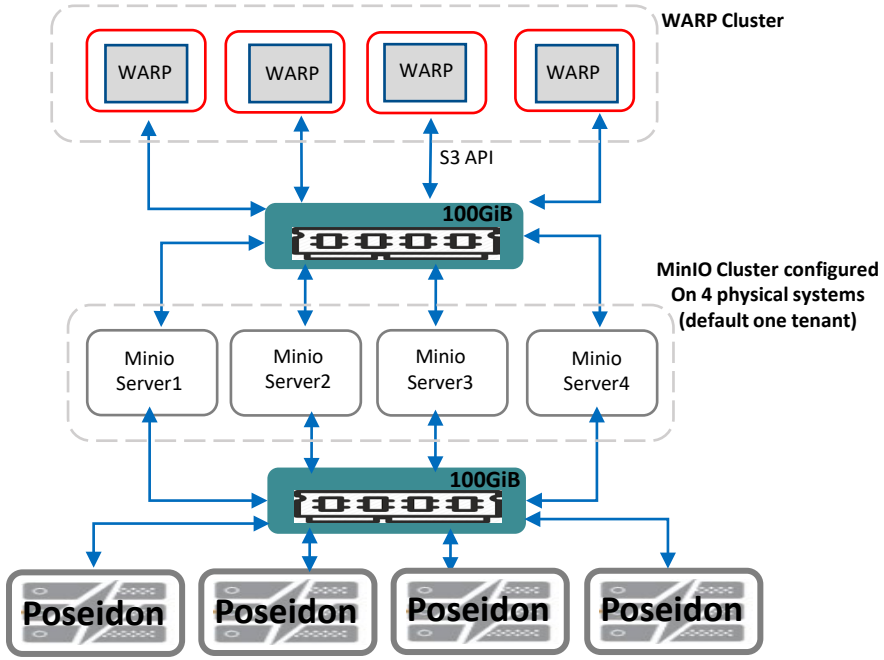
https://min.io/ (open source under GNU AGPL v3)
https://github.com/minio/operator (open source under GNU AGPL v3)
https://kubernetes.io/ (open source under Apache License 2.0)

Demo – Building STaaS with Poseidon

WARP Benchmark Setup with 4 Node MinIO Cluster



WARP Distributed Benchmarking*



MinIO Cluster with WARP benchmark setup on Poseidon

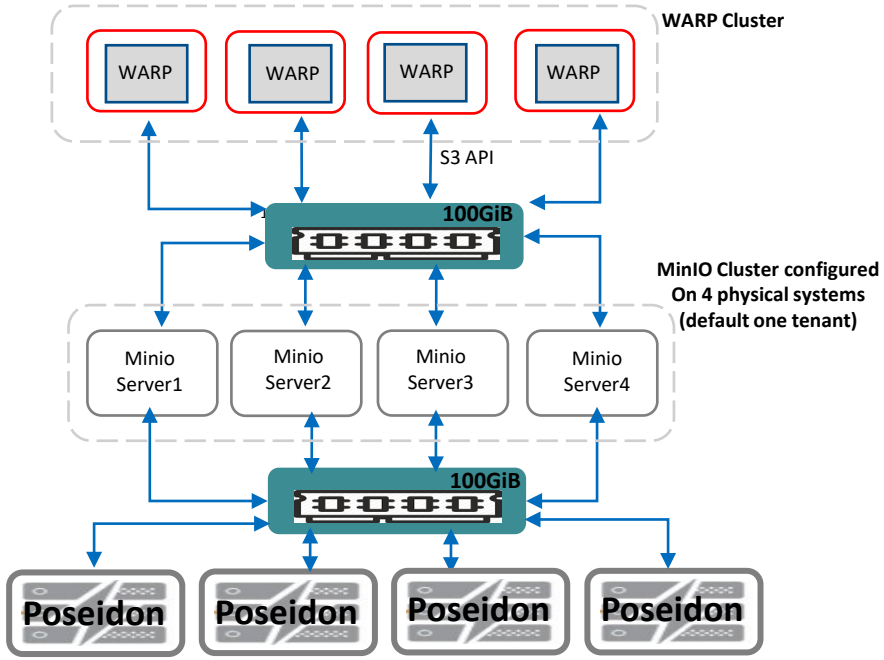
Performance numbers

- Theoretical Max Numbers:
  READ : Theoretical max 50 GB/sec ([4 Nodes* 100 Gig]/8 = 50GB/sec)
  WRITE: Theoretical max 25 GB/sec ([4 Nodes* 100 Gig]/8 /2= 25GB/sec, Erasure coding EC:4)

- Expected Numbers (considering the 80% to 90% efficiency achievable due to overhead**):
  READ : 40 GB/sec to 45 GB/sec and WRITE: 20 GB/sec to 23 GB/sec

- *https://github.com/minio/warp (open source under GNU AGPL v3)
- ** https://min.io/resources/docs/Supermicro-Cloud-DC-Server-Benchmark.pdf

Monolithic STaaS

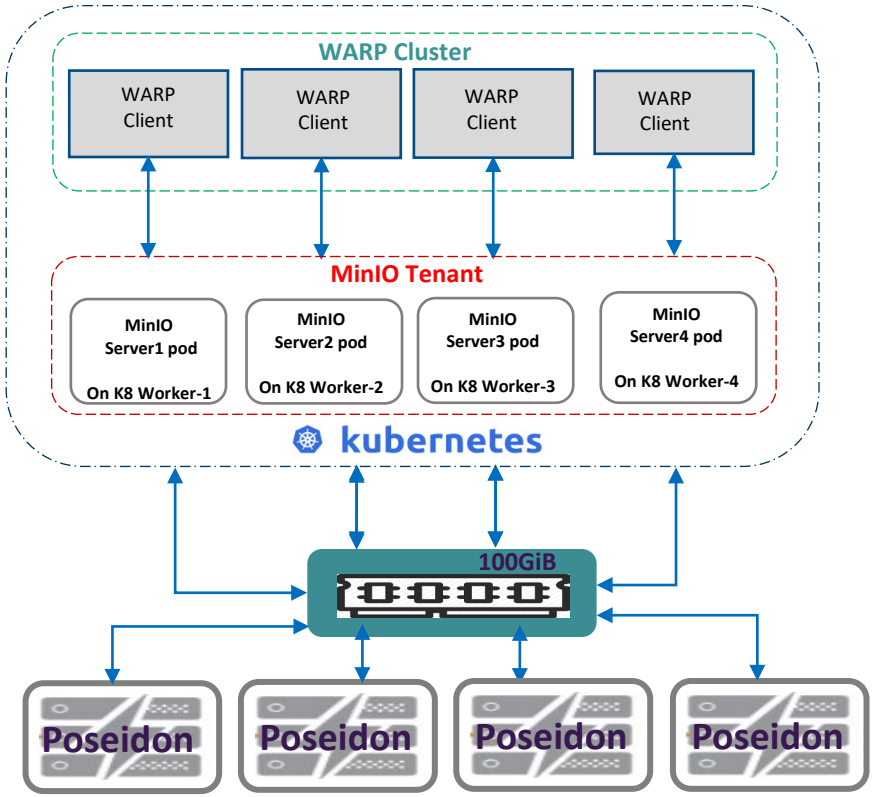


MinIO Cluster with WARP benchmark setup on Poseidon

	GET(MiB/S)	PUT(MiB/S)
WARP-1	10807.9	5843.9
WARP-2	10790.3	5821.93
WARP-3	10746.2	5817.03
WARP-4	10778.1	5789.83
Total	43122.5	23272.7

Numbers obtained with the 4 WARP clients run on 4 MinIO standalone servers.

Cloud Scale STaaS

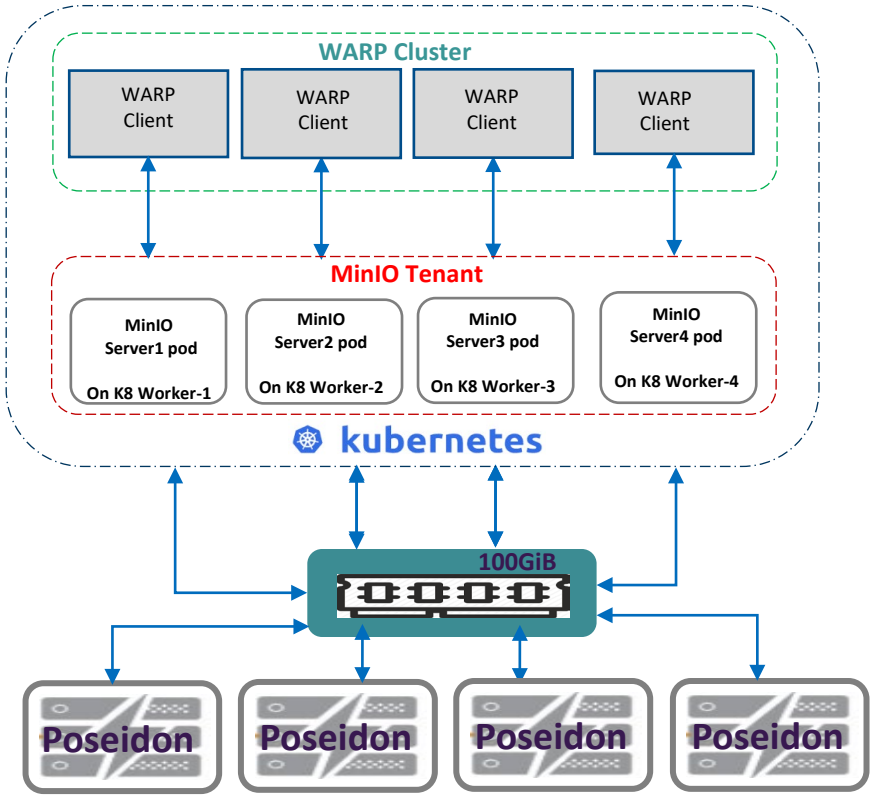


WARP Distributed Benchmarking of STaaS on K8

- ✓ Cloud Native STaaS deployment
- ✓ True Multi Tenancy support
- ✓ Consistency across multiple Tenants

Demo – Running WARP in Kubernetes

Cloud Scale STaaS

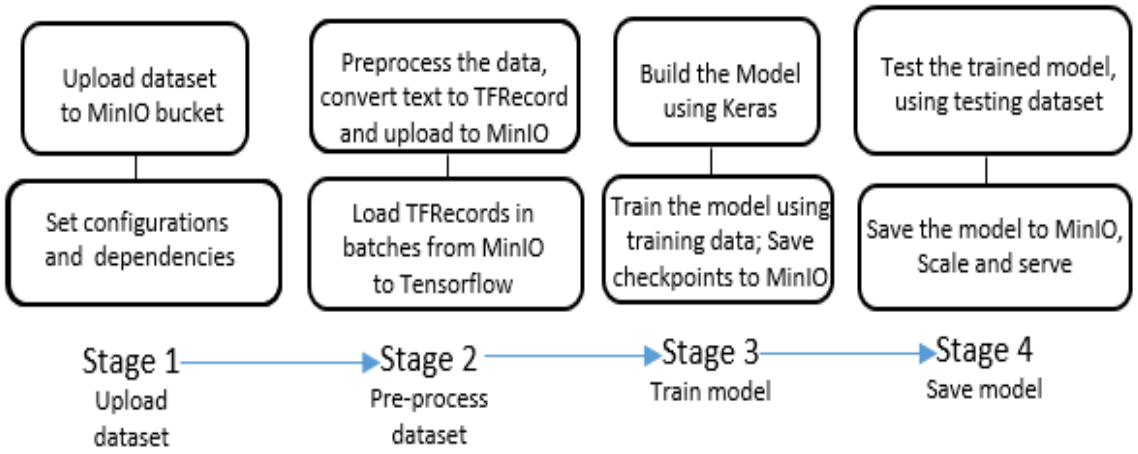


WARP Distributed Benchmarking of STaaS on K8

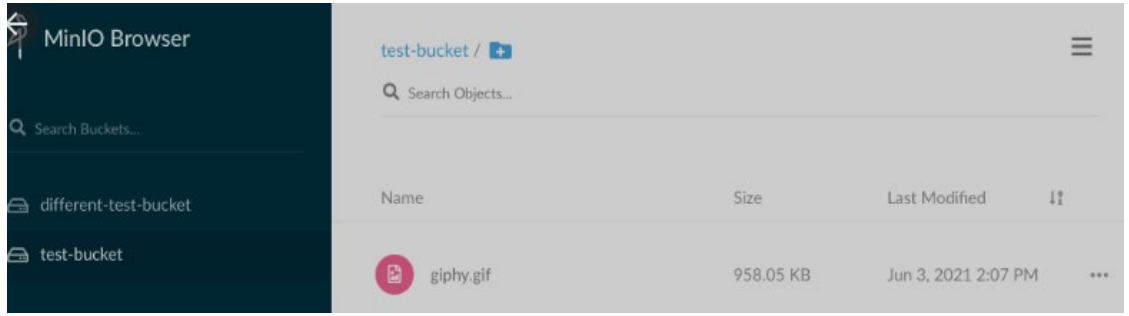
Operation Object Size (in MB)	GET(READ) (in MiB/s)	PUT(WRITE) (in MiB/s)
1	8136.04	3219.6
20	27611.2	13283.65
64	27945.7	16505.66
128	27428.86	20759.45
256	28950.94	22909.15
512	28698.06	23165.82
1024	29858.72	23336.27

Numbers obtained with the Kubernetes 4 MinIO servers with the 4 MinIO WARP clients.

Deploying a 4 Stage AI/ML Pipeline on STaaS Solution



- A MinIO bucket carved out of a MinIO tenant



- We deployed a ML workload by integrating a four stage machine learning pipeline on the MinIO tenant
- Each stage of the pipeline interacts with MinIO and loads and stores the desired data on-demand

- Script snippet referring to MinIO bucket

```
namespace = "my-ml"
random_seed = 44
batch_size = 128
datasets_bucket = "datasets"
preprocessed_data_folder = "preprocessed-data"
tf_record_file_size = 500


# How to access MinIO
minio_address = "minio-svc.minio.svc:9000" #Server IP found from pod
minio_access_key = "minio" #default
minio_secret_key = "minio123" #default

import os
import random
import tarfile
import timeit
from datetime import datetime
```


Conclusion

- With the growing number of cloud native workloads and applications, it is essential for storage to be cloud native as well.
- In this presentation, we have demonstrated a low cost cloud scale STaaS deployment on Poseidon Storage, based on all open source technologies (MinIO, Kubernetes).
- Our Kubernetes STaaS deployment gives comparable performance as monolithic object based STaaS deployments, with all the additional features and benefits provided by the cloud.

Acknowledgements

- Madan Udaykumar
- Byju Ravindran
- Sathish Kumar. M

Thank you!

Our Contact:


s.chawdhary@samsung.com


abdul.amir@samsung.com


sandeep.agar@samsung.com


junho4.jang@samsung.com

Please take a moment to rate this session.

Your feedback is important to us.

