KubeCon | CloudNativeCon

North America 2017

Economics of using Local Storage Attached to VMs on Cloud Providers

Pavel Snagovsky, Software engineer, *Quantum Corp*

Agenda

- Persistent storage options to support Kubernetes workloads on public clouds
- Example of utilizing storage orchestrator backed by “local storage” with demonstrated benefits
- Storage beyond persistent volumes
- Conclusions and Q&A

Persistent storage in public cloud

Cloud providers offer very flexible persistent disks/elastic block storage (EBS)

- Highly durable
- Snapshots
- Performance
- Elasticity
- Encryption
- Dynamic provisioning

Persistent storage in public cloud (cont.)

Persistent block store offerings are falling short to meet the needs of modern cloud native applications

- Slow provisioning and failover times
- Expensive
- Zone locality limitations
- Proprietary

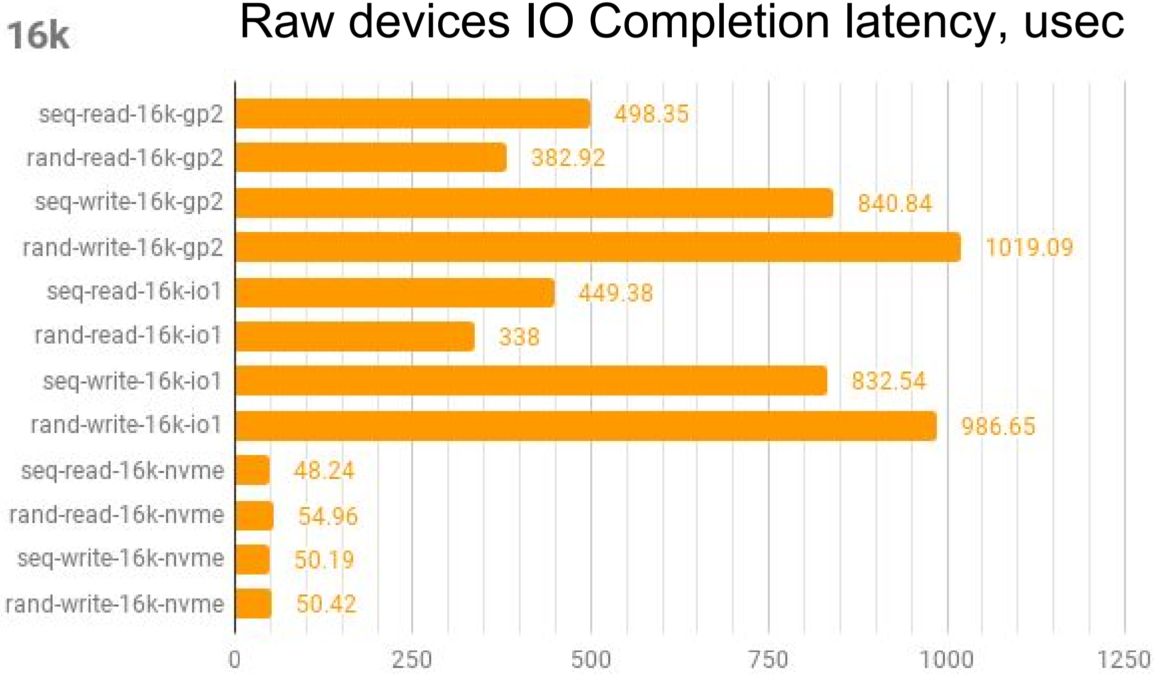
Local storage

GCE and AWS offer instances with local storage

- Pros
    - Low latency, high performance
    - Inexpensive
    - Transactional and streaming IO options (AWS)
    - Consistent IO performance
- Cons
    - No dynamic provisioning (AWS)
    - Data and durability tied to the node

Instance store vs. EBS (AWS)

Raw devices IO Completion latency, usec



Harnessing local storage

Local storage delivers solid performance at low cost.

Problem:

- Durability, locality tied to instance
- Lifecycle Management

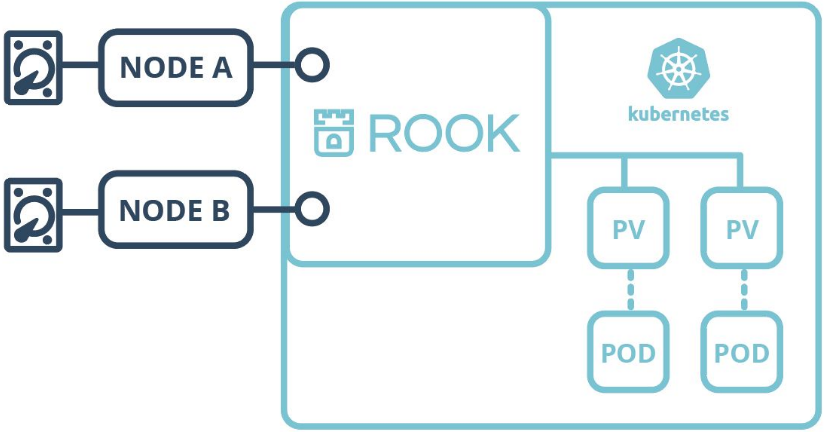
By solving these problems we can capitalize on the benefits of cloud instances local storage.

Harnessing the benefits with Rook

Rook overview

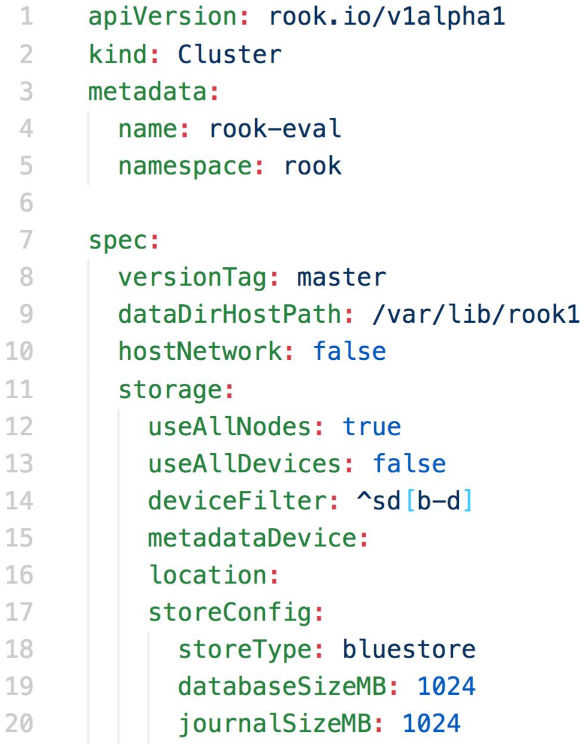
- Leverages Kubernetes features, Operator, CRDs, StorageClass, PV
- Backed by Ceph
- Fully automated lifecycle management of underlying storage
- Self healing and monitoring
- Block, file and object store

https://rook.io



Rook High level architecture

Running Rook in a cloud



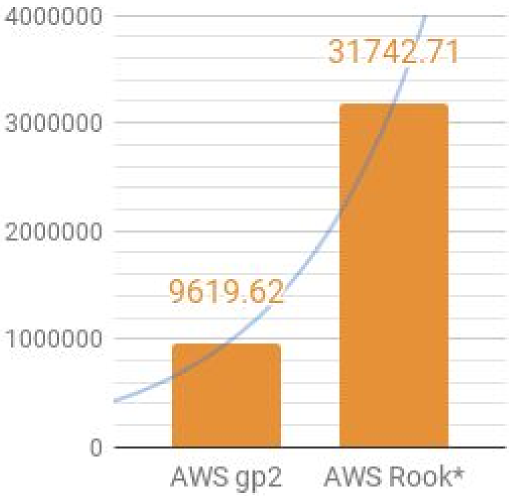
```yaml
apiVersion: rook.io/v1alpha1
kind: Cluster
metadata:
  name: rook-eval
  namespace: rook

spec:
  versionTag: master
  dataDirHostPath: /var/lib/rook1
  hostNetwork: false
  storage:
    useAllNodes: true
    useAllDevices: false
    deviceFilter: ^sd[b-d]
    metadataDevice:
    location:
    storeConfig:
      storeType: bluestore
      databaseSizeMB: 1024
      journalSizeMB: 1024
```

DEMO

Performance (AWS)

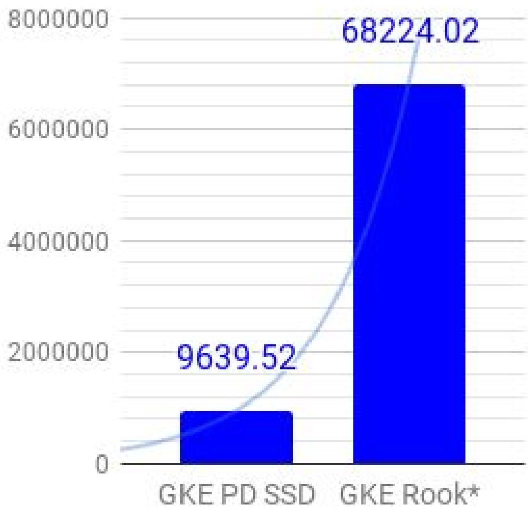
Read IOPs 16K block size as measured from a test pod with FIO.



- Storage backed by instance local disks is significantly more performant (particularly for random reads) than persistent volume offerings of public cloud providers.
- Write performance depends on the type of replication and impacted by network throughput.

Performance (GKE)

Read IOPs 16K block size as measured from a test pod with FIO.



- Storage backed by instance local disks is significantly more performant (particularly for random reads) than persistent volume offerings of public cloud providers.
- Write performance depends on the type of replication and impacted by network throughput.

Pod failover

Typical pattern of dynamically provisioning persistent volumes for a pod result in a pod failover times of **1-5 minutes** (with EBS) due to:

- Device attach/detach penalties
- API calls
- Locality constraints and resource dependencies
- Stuck volumes

Pod failover comparison

With Rook there is none of these penalties and constraints. From the moment scheduler decided to reprovision the pod it takes **milliseconds to a second** for a pod to start with mounted volume.

Rook failover is instant!

Multi AZ considerations

- Kubernetes should be be running multi AZ for cluster resiliency
- Persistent disk is available only within a single zone, moving it to a different zone can be time consuming

Zone failover is not possible when using persistent disk/EBS.

Compatibility

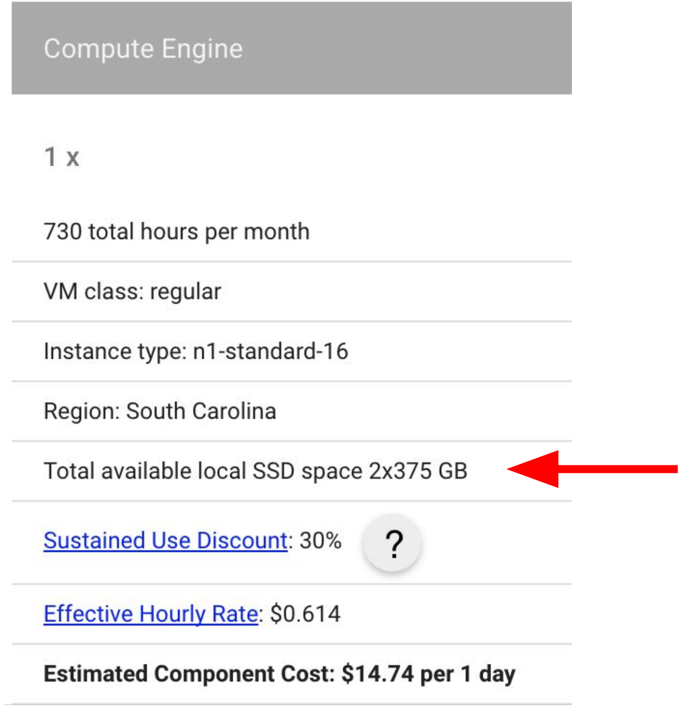
Rook runs everywhere Kubernetes runs

Storage orchestrator such as Rook abstracts storage layer making the solution highly portable for Kubernetes users.

- Portable code
- No vendor lock-in
- Multi provider environments
- Testing on any environment
- Avoiding the dependence on provider specific functionality

Kubernetes administrator would have to setup a storage cluster, define StorageClass, the user managed resources can remain the same for any environment.

GCE local storage costs



Compute Engine

1 x

730 total hours per month

VM class: regular

Instance type: n1-standard-16

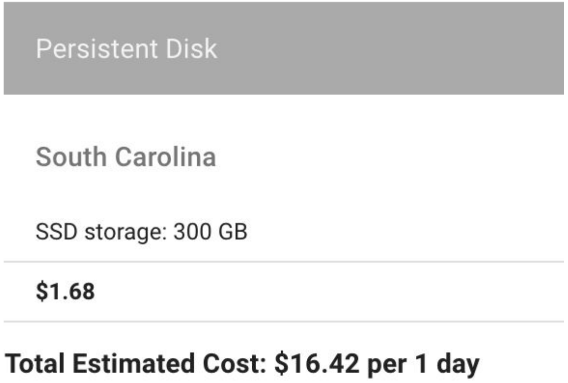
Region: South Carolina

Total available local SSD space 2x375 GB

Sustained Use Discount: 30% ?

Effective Hourly Rate: $0.614

Estimated Component Cost: $14.74 per 1 day



Persistent Disk

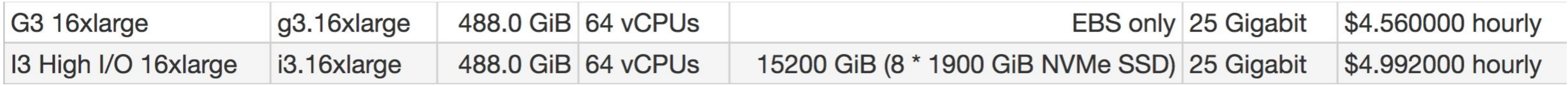
South Carolina

SSD storage: 300 GB

$1.68

Total Estimated Cost: $16.42 per 1 day

AWS local storage costs



G3 16xlarge	g3.16xlarge	488.0 GiB	64 vCPUs	EBS only	25 Gigabit	$4.560000 hourly
I3 High I/O 16xlarge	i3.16xlarge	488.0 GiB	64 vCPUs	15200 GiB (8 * 1900 GiB NVMe SSD)	25 Gigabit	$4.992000 hourly

- G3 16xLarge
  - 488GiB Ram
  - 64 vCPUs
  - EBS Only
  - 25GB Networking
- $4.56/hr

- I3 High I/O 16xLarge
  - 488GiB Ram
  - 64 vCPUs
  - 15,200GiB (8 NVMe) SSD
  - 25GB Networking
- $4.992/hr

Costs Summary

- The cost of storage devices attached to the instance is minimal, disk is cheap, you are mostly paying for Compute resources that are available for utilization within your Kubernetes cluster.
- Instance storage is generally much cheaper than persistent block storage
- Slicing the bigger pool of storage will allow more efficient utilization.

Use cases

Combine different approaches to meet different performance requirements and reduce costs.

- Use Rook object store instead of cloud provider object store service
- Multiple clusters and/or storage pools to meet different performance requirements and provide isolation
- Use persistent volumes from your cloud provider as a backend devices of your storage orchestrator
- Save by switching to in-cluster services over the provider equivalents, such as databases - `$ helm install --name my-release stable/mysql`

Conclusion

ROOK backed by Cloud provider ‘local’ disks deliver better cloud native experience in terms of

- Compatibility and Kubernetes integration
- Performance
- Provisioning and failover times
- Cost

Questions

Pavel Snagovsky

pavel.snagovsky@quantum.com

GitHub: @paha

We are hiring. Rook on!

Learn more about Rook - https://rook.io

FIO test results and related files

RE: https://github.com/paha/rook-aws