Scalability Study of Seagate's Distributed System: CORTX on Kubernetes
Zeyuan (Faradawn) Yang
June 2022

Part 1: 8-node scalability study

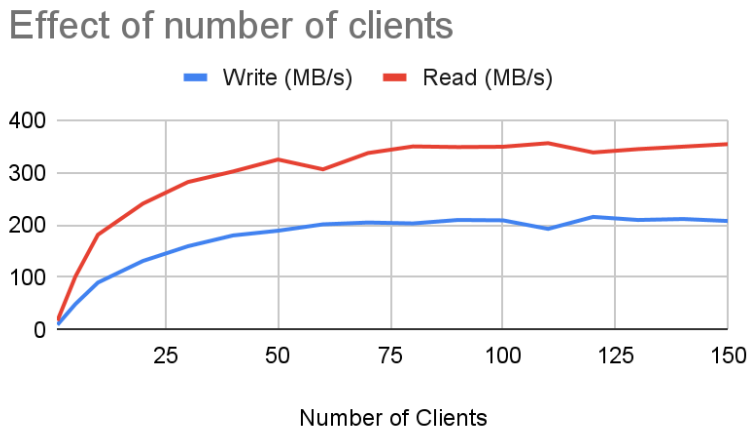
A. Testing parameters

- CORTX v0.6.0 on Kubernetes v1.24.
- Eight Chameleon storage nodes at CHI@TACC with CentOS 7.9,2009.
- CORTX data pod size: 5G x2. Metadata pod: 5G x1
- Tested 5000 samples, each 1MB
- Writing took 30s, Reading 15s, Deleting 5min. 7 experiments took 1 hour
- Maybe can enlarge pod size and skip delete

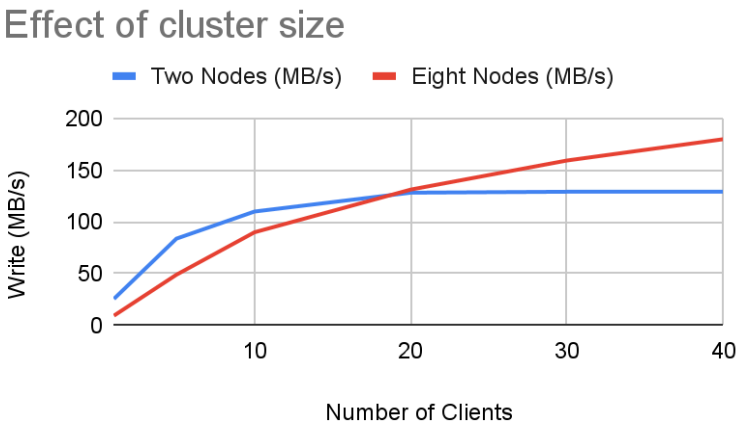
B. Conclusions

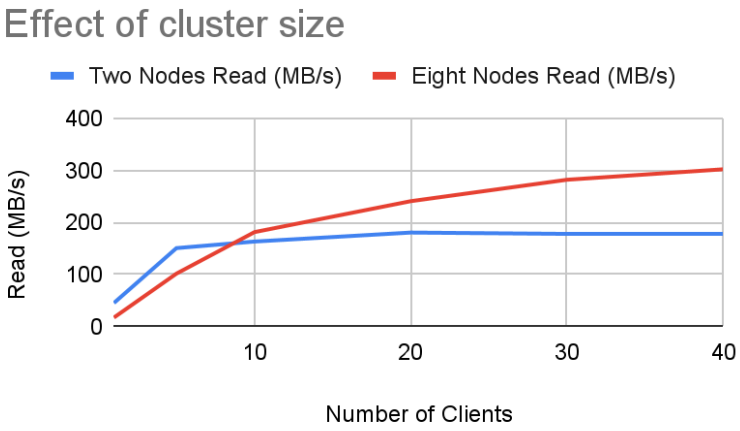
- <u>One: the number of samples has little effect on speed</u>
    - Once passed the 5000-sample (5 Gigabytes) threshold, the number of samples has little effect on read/write speed. (10 clients)

- <u>Two: the number of clients has a positive effect on speed</u>
    - Once passed the 50-clients threshold, the throughput plateaued at 350 MB/s for Read and 200 MB/s for Write. (5000 samples)



- Three: the size of CORTX cluster has a positive effect on on speed
    - Tested a two-node and eight-node CORTX cluster
    - When the number of clients is below 10, the two-node cluster performs better. Perhaps, the load-balancing across eight nodes costs time.
    - However, when the inflow of clients becomes large, the eight-node cluster outperforms the smaller one. (5000 samples)
    - Future
        - One node, two and six node, powers of two
        - Extend to 80 number of clients
        - Motr testing with S3 layer
        - IO benchmark tool: HPC IOR
            - Has S3 interface, (Ceph - rados, CORTX - motr)
            - Modify rados backend, to measure S3 motr
    - Why two-node faster?
        - Eraser coding?





S3Benchmark output:

```
Parameters:
  label:                  2022-6-9-14-35-30-535857360
  numClients:             30
  objectSize (MB):        1.000
  copies:                 0
Tests:
  Operation:              Write
  RPS:                    159.574 (request per second)
  Total Requests Count:   5000
  Errors Count:           0
  Total Throughput (MB/s): 159.574
  Total Duration (s):     31.333
  Total Transferred (MB): 5000.000
  Duration Max:           0.913
  Duration Avg:           0.187
  Duration Min:           0.080
  Ttfb Max:               0.913
  Ttfb Avg:               0.187
  Ttfb Min:               0.080
  Duration 90th-ile:      0.276
  Duration 99th-ile:      0.596
  Ttfb 90th-ile:          0.276
```

Ttfb 99th-ile: 0.596

Operation: Read
RPS: 282.400
Total Requests Count: 5000
Errors Count: 0
Total Throughput (MB/s): 282.400
Total Duration (s): 17.705
Total Transferred (MB): 5000.000
Duration Max: 0.400
Duration Avg: 0.106
Duration Min: 0.043
Ttfb Max: 0.398
Ttfb Avg: 0.105
Ttfb Min: 0.042
Duration 90th-ile: 0.136
Duration 99th-ile: 0.180
Ttfb 90th-ile: 0.135
Ttfb 99th-ile: 0.178

Summary of two papers

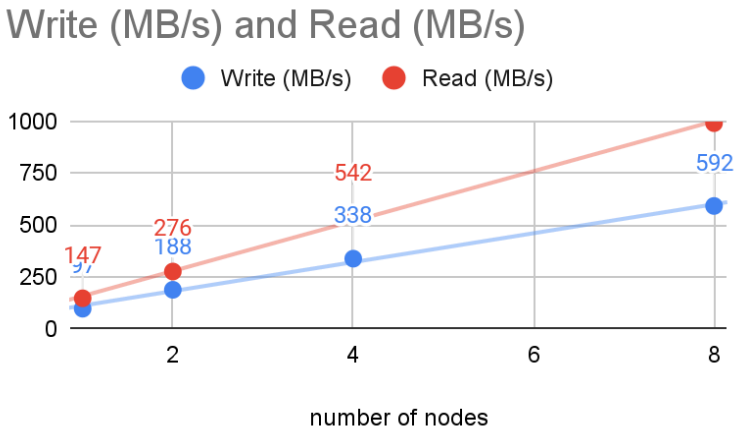
- *Principled Schedulability Analysis using TAM (Remzi)*
    - Problems to solve
        - Scheduling didn't consider the weight of clients
        - Unbounded read latency
        - Lack local scheduling control point
    - What researchers invented
        - Developed TAD alyzer to produce graphs to identify in which stage there is a scheduling deficiency
        - Identity scheduling problem without knowing low-level implementation detail
- *Deconstructing Commodity Storage Cluster (Haryadi)*
    - What is the current problem
        - Commodity storage systems only offer application-level interfaces that hide the complex internal structure.
        - Want probe the underlying system
    - What researchers invented
        - A method to gauge EMC's policies, write update protocol, caching, replication, load-balancing
        - Instrument each standard component
        - Achieved by tracing disk and network traffic
    - Method
        - Passive observation: install a software into kennel to monitor read/write ro relay message
        - Delay: delay message A, see what subsequent messages, B, C, D, are delayed.

Part 2: 16-node scalability benchmark

Raw data at this Google Sheet

One: whether the number of nodes affects throughput: yes

- Nodes vary, 2 pods, 64Gi, nreq 100, objsize 16M, nclients 80 (total payload 128G)
- The top right red dot is 991 MB/s

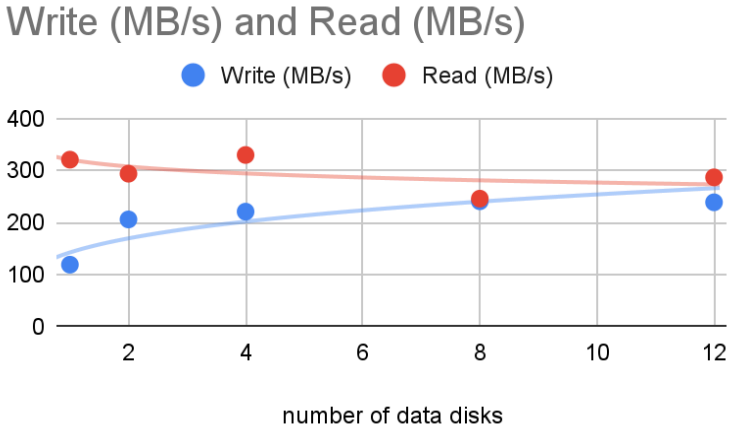


Two: whether the number of data disks affects throughput: no

- 8 nodes, disks vary, 64Gi, nreq 100, objsize 1M, nclients 80
- In this solution.yaml, we used two “data disks”.

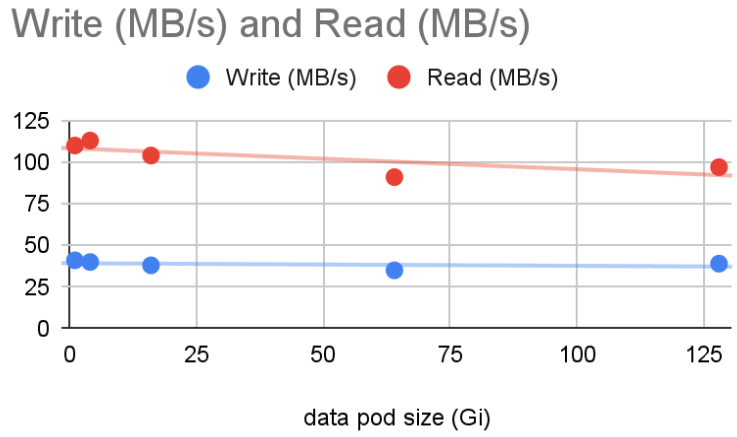
```
171   storage:
172     cvg1:
173       name: cvg-01
174       type: ios
175       devices:
176         metadata:
177           device: /dev/sdc
178           size: 5Gi
179         data:
180           d1:
181             device: /dev/sdd
182             size: 5Gi
183           d2:
184             device: /dev/sde
185             size: 5Gi
```

-



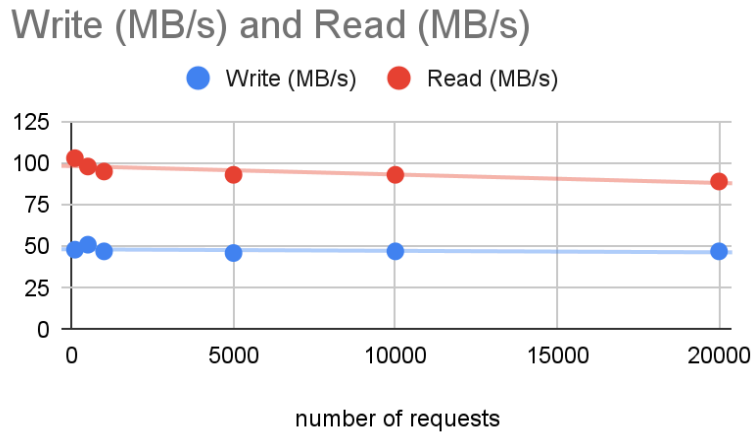
Three: whether the data disk size affects throughput: no

- 8 nodes, pods 2, size vary, nreq 100, objsize 1M, nclients (5)



Four: whether the number of requests affects throughput: no

- 1 node, 2 pods, 64Gi, nreqs vary, objsize 1M, nclients 5



Five: whether object size influences throughput: yes

- 8 nodes, 2 pods, 64Gi, nreqs 100, objsize vary, nclients 80
- Object size: powers of two

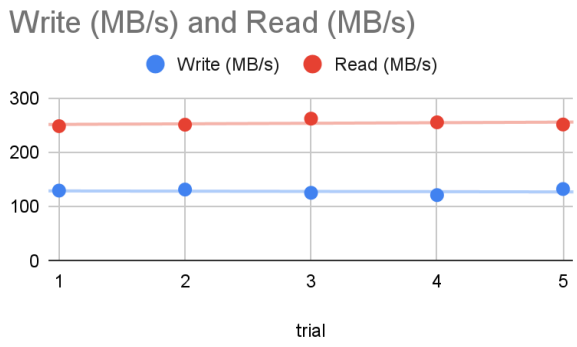
Six: whether number of clients affects throughput: yes

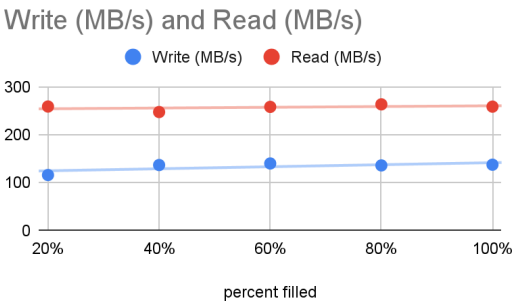
8 nodes, 2 pods, 4Gi, nreqs 100, objsize 16M, nclients vary

Seven: whether throughput decreases as the storage fills up

- 8 nodes, 2 pods, 5Gi, nreq 100, objsize 1M, nclients 20
- First, each time upload 2Gi; 5 times; delete the objects after each upload.
- Second, 2Gi 5 times, but without deletion. So, the storage gradually fills up: 2Gi, 4Gi, 6Gi, 8Gi, 10Gi (full).
- Expected read speed to dwindle as the storage filled up. However, the read / write speed seemed unaffected by the percentage filled.

- [Future] Test fragmentation
    - Fill once with different object size
    - Delete 50% randomly (create random holes in block allocation)
    - Fill second time
    - Delete 50%
    - Are there HDD nodes on Chameleon





Part 3: Tutorial Videos on Deploying CORTX

Part 1: How to create an instance on Chameleon: https://youtu.be/AVc0MUXeycU
Part 2: How to install Kubernetes: https://youtu.be/s-TsYbFl5dA
Part 3: How to deploy and benchmark CORTX: https://youtu.be/6E5K0z910y4