Stress testing Object Stores

Alastair Dewhurst

Introduction

- Cloud meeting on 5th July, slides from Peter and Jarka:
  - https://indico.cern.ch/event/650137/
  - Decouple object stores and Event Service pending a better understanding of object store performance and limitations.
  - Want to advise Rucio/ES on best practices when using object stores.
  - Provide a tool for site admins to mimic ADC usage.
- In the longer term, we would also want functional tests for blacklisting purposes.
- All of the discussion so far focused on developing jobs run via Hammer Cloud to do this.
  - Is there something better?

COSBench

- COSBench stands for Cloud and Object Store Benchmark.
  - https://github.com/intel-cloud/cosbench/
- Developed by Intel and used by the likes of RedHat, BBC, Dell, Seagate etc.
  - It is THE object store benchmarking tool.
- Not only can it test the object stores we currently have access to, but also any we might be offered in future.

Table showing the different Cloud Storage API that are supported by COSBench.

Cloud Storage	Auth	Status
OpenStack* Swift	tempauth	OK
	swauth	OK
	keystone	OK
	direct	OK
Amplidata* Amplistor	none	OK
	basic/digest	Verifying
Ceph	librados	OK
	rados GW (swift)	OK
	rados GW (s3)	OK
Amazon* S3	integrated	OK
Scality	sproxyd	OK
CDMI		
CDMI-base	basic/digest	OK
CDMI-swift	swauth/keystone	OK
None	none	OK
Mock	mock	OK

COSBench

Web console or CLI to submit tests.

Controller is responsible for coordinating drivers to collectively execute a workload, collecting and aggregating runtime status or benchmark results from driver instances, and accepting workload submission.

Driver is responsible for workload generation, issuing operations to target cloud object stores and collecting performance statistics.

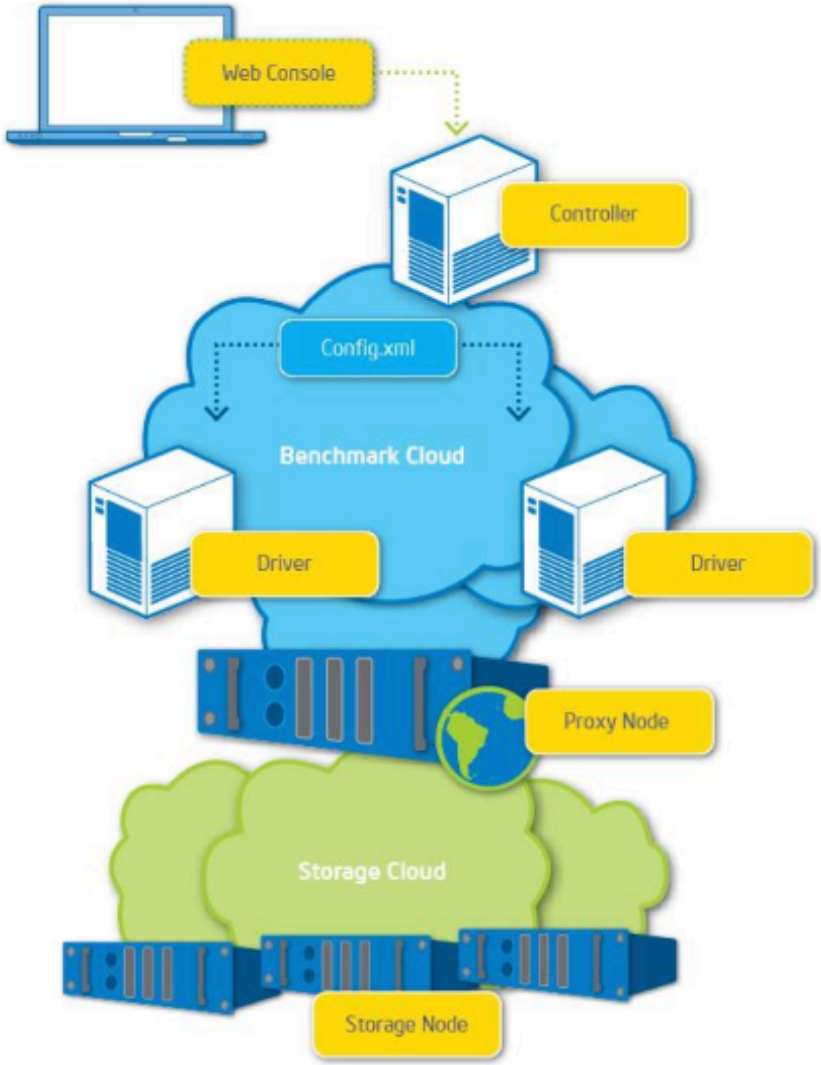


Image taken from: https://github.com/intel-cloud/cosbench/blob/master/COSBenchUserGuide.pdf

Setup

- It was super easy:
  - Took minutes to setup on an SL7 VM.
  - Hardest part was getting Java to recognize Grid CAs.
- I ran some tests to understand how COSBench works.
  - It was on VM infrastructure so as to not accidentally break production work while I figured things out.
    - Look at the type of output not the actual numbers!

XML configuration

- Tests are configured using an .xml file. Two parts:
  - Site configuration - host, secret/access key, bucket name, auth mechanism etc.
  - Test configuration - Size of objects, length of test, fraction of reads/writes/deletes etc.
    - Multiple different tests can be run in series in one xml.
- Average object store file size (from Peter's analysis) is 2 ± 1 MB.
  - Write, Read and 50:50 Read/Write test?
- Can provide any prospective cloud provider with .xml and they can run it themselves.

Test Outputs (1)

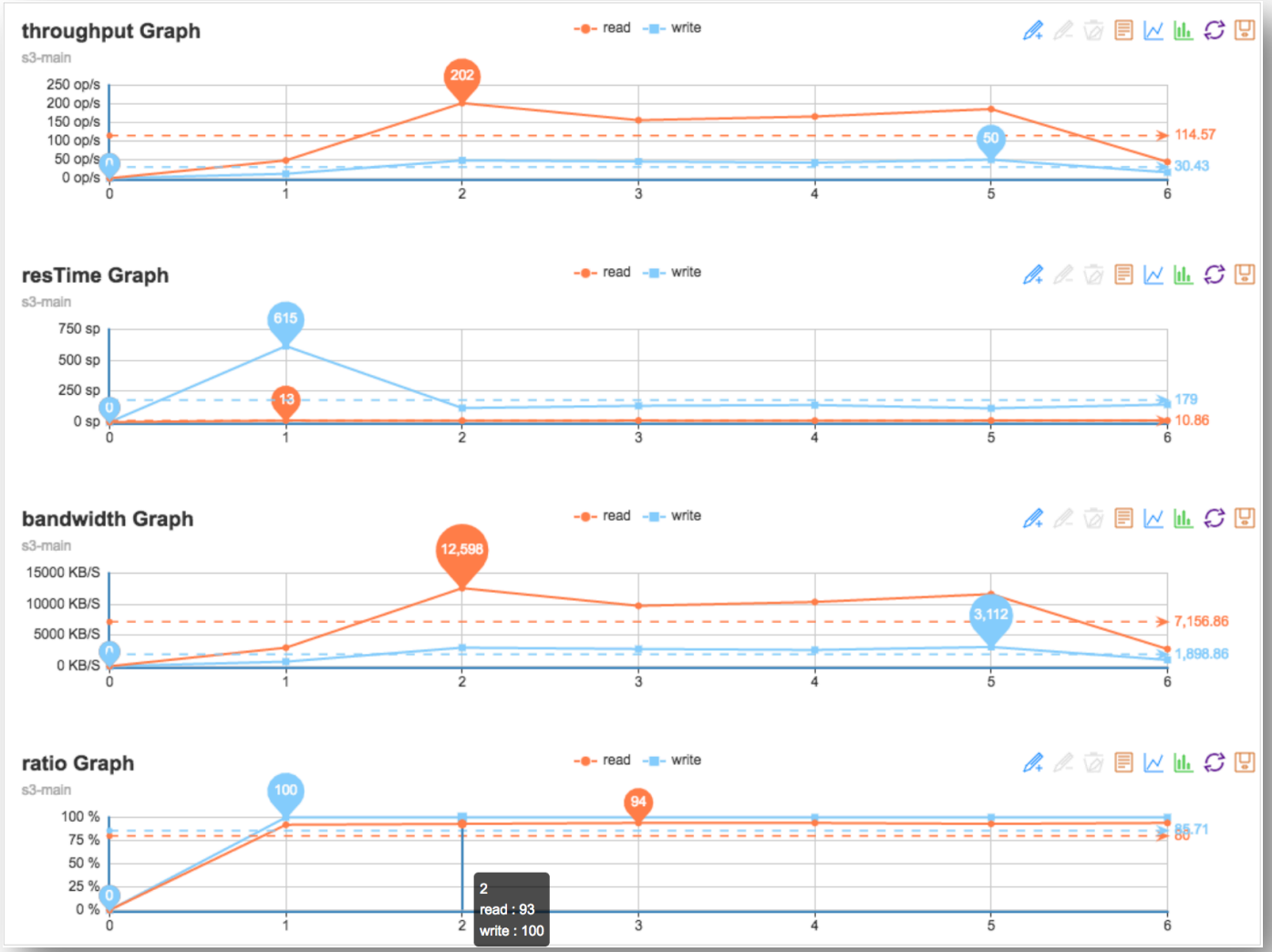
Op-Type	Op-Count	Byte-Count	Avg-ResTime	Avg-ProcTime	Throughput	Bandwidth	Succ-Ratio
op1: init -write	0 ops	0 B	N/A	N/A	0 op/s	0 B/S	N/A
op1: prepare -write	20 ops	1.28 MB	78.3 ms	71.4 ms	12.76 op/s	816.33 KB/S	100%
op1: read	4.21 kops	269.25 MB	12.53 ms	10.28 ms	140.83 op/s	9.01 MB/S	93.03%
op2: write	1.08 kops	69.06 MB	168.18 ms	165.89 ms	36.12 op/s	2.31 MB/S	100%
op1: cleanup -delete	40 ops	0 B	35.82 ms	35.82 ms	27.89 op/s	0 B/S	100%
op1: dispose -delete	0 ops	0 B	N/A	N/A	0 op/s	0 B/S	N/A

ResTime (RT) Details

Op-Type	60%-RT	80%-RT	90%-RT	95%-RT	99%-RT	100%-RT
init-write	N/A	N/A	N/A	N/A	N/A	N/A
prepare-write	< 70 ms	< 110 ms	< 170 ms	< 170 ms	< 170 ms	< 240 ms
read	< 20 ms	< 20 ms	< 20 ms	< 20 ms	< 30 ms	< 210 ms
write	< 80 ms	< 200 ms	< 320 ms	< 550 ms	< 1,050 ms	< 6,760 ms
cleanup-delete	< 20 ms	< 20 ms	< 90 ms	< 200 ms	< 230 ms	< 290 ms
dispose-delete	N/A	N/A	N/A	N/A	N/A	N/A

- A summary table is provided for each test.
  - All test outputs can be exported as .csv.
  - Logs of entire tests are available.

Test Outputs (2)



Next Steps

- One of Jarka's suggestions was to run the tests directly from the HC cluster.
  - Setup COSBench on this cluster?
  - Need to find a way to pull object store info from AGIS / Panda Server
    - Is the secret/access key only stored in Panda?
- Then we would benchmark all current object stores.
  - RTT time to CERN is different for each site but otherwise test would be the same.

Future Work

- At some point we would want to submit functional tests jobs to Panda queues that would test the relevant object store (using Rucio).
- For stress testing from any site to any object store, I think it is still an open question, how best to proceed.
  - COSBench could be easily put into CVMFS
  - Pilots could download the secret/access key and run a COSBench then upload output .csv.
  - Alternatively writing scripts to run many rucio commands might be simpler.

Conclusions

- COSBench is an industry standard tool that is easy to setup.
  - We can also use it to test storage we haven't developed any Rucio tools for yet.
- COSBench will provide an independent way to validate both object stores and Rucio.

Backup

Backup (Jarka's slide)

HammerCloud and ObjectStore commissioning: Ideas

- HCe can stress/functional test ObjectStores
  - We know how to inject jobs to PanDA, can issue requests to Rucio
- Scenario #1
  - DougB was stress-testing OSes in the past
  - Reproduce this with HC
  - The load would originate from HC cluster, go through Rucio to ObjectStores
- Scenario #2
  - Submit jobs to PanDA that issue requests to Rucio
  - The load would originate from PanDA jobs running on WNs around the world
- Scenario #3
  - Submit jobs to PanDA that emulate what ES does
  - The load would originate from PanDA jobs running on WNs around the world, can be useful for ES commissioning

Backup (Peter's slide)

Bucket statistics

- Filtered with June 2017 && EventService_premerge

	count	object size	total size
CERN	620K	1.6±1.0M	300GB
LANCS	200K	1.8±0.5M	360GB
MWT2	3.4M	1.8±1.0M	5.9TB

- PUT tests should mimic these distributions

Backup (Setup details)

- Full instructions can be found at:
  - https://github.com/intel-cloud/cosbench/blob/master/COSBenchUserGuide.pdf

Setup instructions that worked on SL7:

```
# yum install java-1.7.0-openjdk
# yum install nc
# wget https://github.com/intel-cloud/cosbench/releases/download/v0.4.2.c4/0.4.2.c4.zip
# unzip 0.4.2.c4.zip
# sh start-all.sh
```

Trusting CA Certificates (Repeat for all needed CAs):

```
# cd /etc/grid-security/certificates/
# openssl x509 -in UKeScienceCA-2B.pem -inform pem -out UKeScienceCA-2B.der -outform der
# keytool -importcert -alias UKeScienceCA-2B -keystore /usr/lib/jvm/java-1.7.0-openjdk-1.7.0.141-2.6.10.1.el7_3.x86_64/jre/lib/security/cacerts -file UKeScienceCA-2B.der
```

Note: default password is 'changeit'