Introduction

The C3G in collaboration with Calcul Québec has bought the first specialized cloud system (Juno). Note that the current version of the document (July 27 2023) is mainly geared at reviving the mechanics of the cluster and making sure that all its important part is functioning. It will also serve as a basis for improving the performance of the system in the future if deemed useful.

Also note that the most important piece for the C3G, that is running Genpipes on the CephFS has still not happened.

Summary

First all components of the new system, called Juno are functional to at least the level of the old cloud, also called electric secure cloud (esc).

- HPL run's at about 90% of the speed in a Juno VM as compared to the Bare Metal system. This was expected.
- The Block storage has `22 GB/s` of linear read and `6 GB/s` linear write throughput with 200K random IOPS. This is more than the peak usage of Beluga or Narval in a typical week of `~20GB/s` read `~3.5GB/s` write and `~60K` random IOPS.
- The object store saturates at 2GB/s. The bottleneck is the firewall it in front of its API. Note that the firewall is about 100x the speed of the commercial link to it and 2x speed toward Universities and Hospitals.
- The object store metadata server used for example to list object in a bucket is very slow and needs to be tweaked.
- The CephFS is working but is not tested
- The local SSD are working but are not tested

Running HPL

I have been running an optimized version of HPL to be able to compare different configurations of the Compute nodes.

I am using the AMD HPL bliss package, here is the HPL.dat and OMP configuration that I use:

```
HPLinpack benchmark input file
Innovative Computing Laboratory, University of Tennessee
HPL.out      output file name (if any)
6            device out (6=stdout,7=stderr,file)
1            # of problems sizes (N)
150000      # Ns  86000  ~ 64 GB, 125000 ~ 128 GB, 250000 ~ 512 GB
1            # of NBs
```

```
220          NBs
0            PMAP process mapping (0=Row-,1=Column-major)
1            # of process grids (P x Q)
8            Ps
8            Qs
16.0         threshold
1            # of panel fact
2            PFACTs (0=left, 1=Crout, 2=Right)
1            # of recursive stopping criterium
4            NBMINs (>= 1)
1            # of panels in recursion
2            NDIVs
1            # of recursive panel fact.
2            RFACTs (0=left, 1=Crout, 2=Right)
1            # of broadcast
2            BCASTs (0=1rg,1=1rM,2=2rg,3=2rM,4=Lng,5=LnM)
1            # of lookahead depth
1            DEPTHs (>=0)
1            SWAP (0=bin-exch,1=long,2=mix)
64           swapping threshold
0            L1 in (0=transposed,1=no-transposed) form
0            U  in (0=transposed,1=no-transposed) form
1            Equilibration (0=no,1=yes)
8            memory alignment in double (> 0)
```

Here are the comparative value of HPL on a Narval Node vs a SD4H VM node:

Theoretical	Narval	SD4H
2.5 Gflops/s	2.0 GFlops/s	1.8 GFlops/s

Note that the Narval and SD4H VM nodes are exactly the same except for the Ethernet vs Infiniban interconnect. The main thing here is that we are 10% slower than the bare metal system which is what we were expecting.

I also ran HPL for fun on 40 CPU nodes with `P = 40`, `Q = 64` and `N = 948683`. It helped us find a problem with the networking connection that we were able to fix and we reached ~40 TFlops/s. This is to say that on our 50 GB Ethernet connection without tweaking much the mpi libraries or the HPL test parameters, and in virtual space, with only 2560 cores, we would have topped the top 500 HPC chart... in 2004!

SSD Block Storage

We have pretty good numbers on the block storage, especially for IOPS, reaching 1,4 million IOPS in read and 100K IOPS in write. For the throughput the

numbers are also pretty good and are comparable or better than the load of the Narval lustre system as a whole, which should be more than enough for us.

	sequential io (GB/s) R W	IOPS (k/s) R W WR
1 Node	4.7 1.5	67 40 49
SATURATION With many nodes	22.1 6.0	1400 100 200

Then the question, what is throttling the RDB cluster. The internet link of `50Gb/s` for the compute nodes, the ceph nodes or the spines are not saturated, the IOPS and throughput of individual SSDs on the ceph side still has some room. While the CPU at max read and max write goes only up to 80%, it still seems that it where the bottle neck is, that is in line with information found on web benchmark and from consultants we had talked to before the purchase. There is still tweaks that can be done to get more speed on the ceph cpu, it has not been tested here. The main tweak are to disable C-state in the bios and have `mitigations=off` in grub. While C-state will only overheat the cpu, `mitigations=off` could be an issue, but since these machine are not shared and only run a *trusted software*, it would probably be reasonable to set it to `off`.

```
todo
Curve on a single node for sequential random and iops against thread on HPC side
Curve on a single node for sequential random and iops against thread on HA side
Curve for the 40 cpu node for sequential random and iops against thread (add HA side?)
```

Throughput config

```
[global]
iodepth_batch_complete_max=64
iodepth_batch_submit=64
group_reporting
verify=0
time_based=1
ramp_time=2s
directory=/tmp
ioengine=libaio
iodepth=64
direct=1
size=1G

[the_test]
runtime=180
rw=<read or write>
bs=4096k
numjobs=64
name=dummy-file
```

iops config

```
[global]
iodepth_batch_complete_max=64
iodepth_batch_submit=64
group_reporting
verify=0
time_based=1
ramp_time=2s
directory=/tmp
ioengine=libaio
iodepth=64
direct=1
size=1G

[the_test]
runtime=180
rw=<write or read>
bs=4096k
numjobs=64
name=dummy-file
```

HDD Object store

We easily reach the firewall limit on the object store `~15Gb/s-20Gb/s` or `~2 GB/s`. The object store API is being software firewall VyOS that cannot give more than right now. However, it seems that VyOS has plans to modify their tooling so it can reach `~40-50 Gb/s`.

Note that this limit will not hold for Globus since the server will have a direct access to the Radow Gateway API without the software firewall.

	sequential io (GB/s) R W
1 Thread	0.05 0.08
1 Node	0.7 0.5
SATURATION behind firewall	2.0 2.0
SATURATION, no firewall (globus only)	? ?

The metadata server part is a bit more problematic. Queries are rather slow, this will need to be tweaked to get some improvement. For example, we need to make sure that the nvme cache is used for the MDS. However metadata access is expected to be slow for the object store. Also, the number of requests per second for a single IP is limited to 10/s on API, which was enough for testing.

Config for fio :

```
[global]
ioengine=http
name=throughput
direct=1
filename=/big-bucket/object
http_verbose=0
https=on
http_mode=s3
http_s3_key=5771db0a65d64b208dddfc10f5723dd2
http_s3_keyid=f1a1ab15f1a5439f8b671f8aa2907829
http_host=objets.juno.calculquebec.ca
http_s3_region=''
group_reporting

# 10G in total, maybe I should use a time setup
[size]
rw=<read or write>
bs=4096k
size=10G
numjobs=64
time_based=1
ramp_time=2s
runtime=180
```

```
todo
Curve on a single node for sequential random and iops against thread on HPC side
Curve on a single node for sequential random and iops against thread on HA side
Curve for the 40 cpu node for sequential random and iops against thread (add HA side too?)
Transfer speed from McGill with S3 (Typical MOH dataset)
Transfer speed from Beluga/Narval with S3 (Typical MOH dataset)
Transfer speed from McGill with Globus (Typical MOH dataset)
Transfer speed from Beluga/Narval with Globus (Typical MOH dataset)
```

SSD CephFS

The cephfs in available on a different network than the

```
todo
Curve on a single node for sequential random and iops against thread on HPC side
Curve on a single node for sequential random and iops against thread on HA side
Curve for the 40 cpu node for sequential random and iops against thread (add HA side?)
```

Local SSD

No tests where ran.

MOH run

How smooth/fast is a *normal* MOH run on the system.

Find the dataset that was used for the test on beluga.