di-EOS - “distributed EOS”:
Initial experience with split-site persistency in a production service

A.J Peters; L Mascetti; J Iven; X Espinal Curull
(all IT-DSS, CERN, Switzerland)
contact: jan.iven@cern.ch

In order to accommodate growing demand for storage and computing capacity from the LHC experiments, in 2012 CERN tendered for a remote computer centre. The potential negative performance implications of the geographical distance (aka network latency) within the same “site” and storage service on physics computing have been investigated. Overall impact should be acceptable, but for some access patterns might be significant. Recent EOS changes should help to mitigate the effects, but experiments may need to adjust their job parameters.

1. Introduction and Background

The CERN Meyrin computer centre (B513) had been built in 1972. Despite regular (including recent) upgrades, the infrastructure is limited to 3.5MW in the electric power it can supply for computing equipment. While the increasing computing needs from the CERN experiments have been acknowledged, initial plans for further extending this capacity inside CERN have been rejected on both technical and capital expenditure grounds. In 2011 this led to a formal Call for Tender for a remote extension to the CERN computing centre, this was won by the “Wigner Research Center for Physics” in Budapest/HU. The contract includes “remote-hands” hardware operations, but all equipment is owned by CERN, and higher-level services at Wigner will be run from CERN.

CERN's EOS disk-only storage service is among those that will have machines in Wigner (see [1] and [2] for an introduction to EOS). Ramp-up of the computing equipment in Wigner will be gradual, with about 10% of CERN's computing and online storage capacity being already installed in autumn 2013, and eventually raising to about half of CERN physics' computing capacity. The new centre had to be seen as an extension of the existing CERN infrastructure, and existing storage services such as EOS would have to run seamlessly across both sites.

The scope of this article is the access from CERN batch clients to CERN online storage on EOS in the light of the increased latency due to the geographical separation of the two sites. While CERN EOS instances are regularly access from external clients (via FTS scheduled transfers or more recently directly from remote batch jobs via XRootD federations), these all expect and deal with WAN-scale latencies. The focus of this article is also on read access as the predominant access from batch machines (and since write is more benign with respect to latency, as will be shown later).

1.1. Business continuity?

A typical benefit from a geographically distant site is “business continuity” in the case of catastrophic failure on one site. Certain CERN service will fully exploit this, but for EOS this is only partially possible – in particular, a complete Meyrin failure will not be transparent to EOS users, and neither will be – to a lesser degree – a full outage at Wigner: due to the “soft” placement and rebalancing algorithms required by the unequal size and growth of the two sites, some files will have replicas only on one site. In addition, the read-only namespace at Wigner will stall writers until Meyrin is back up again (or until manual fail-over). As a consequence, no systematic effort will be made to enumerate and remove all Wigner/Meyrin interdependencies for EOS.

2. Expected impact of network latency on EOS clients

The various interactions between clients and EOS can be classified along different axes: random or streaming, metadata vs data and finally by volume. These parameters and their expected behaviour under increased latency (remote vs local) will be compared to known experiment usage.
Concentrating just on whether access is “remote” or “local”, we have the following classes

• “local”: Meyrin client → Meyrin server: existing setup, used as the reference.
• “local”: Wigner clients → Wigner diskserver – expect slightly worse than the above, since clients may need to contact other machines at CERN first (latency would affect this). Data access then should show comparable performance to the Meyrin - Meyrin case.
• “remote”: Meyrin client → Wigner diskserver: local connection overhead, then full impact of latency on data access
• “remote”: Wigner client → Meyrin server: slowest; both the initial connection setup via other Meyrin-only services as well as subsequent data access suffer from latency.

For “sufficiently large” data transfers in streaming mode and given EOS’ file layout, we assume that single-stream read data rate will eventually be limited by the single disk actually storing the file (EOS scales in the number of concurrent connections to different files, not by providing maximum bandwidth to a single file). The typical LHC experiment data file is in the multi-GB range, and in this case, Linux networking (i.e TCP autoscaling) is efficient enough for the network to not be a limiting factor also for WAN latencies. EOS-internal traffic (such as draining and balancing), as well as a experiment data import/export and batch write activity (MC production, event reconstruction via local temporary files) all are in this category, as is batch read activity for non-direct-I/O (i.e copy to local disk).

For small quantities of data (and metadata-only access), the initial connection setup overhead and authentication should be of a similar dimension as the actual data transfer, so the overall maximum transaction rate will suffer from increased latency on the network path. However, such a usage is not the norm for batch jobs.

Intense metadata operations (namespace dumps, catalogue consistency checks etc) usually can be explicitly placed “close” to the EOS namespace, so the impact of increased latency on final experiment activity will be small. For disk-backed data operations, random seeking on the disk will itself slow the operation rate down, and is of a similar dimension as the network latency – remote access will be slower, but not by orders of magnitude. In this category (besides pure metadata operations), we would have analysis job result file write-back and log file storing (both of which occur typically only once per job), as well as interactive user activity (where latency in the 10s of msec hardly matters).

So far, the latency impact would be relatively benign. However, a worst-case scenario for high latency on the data path would be a sequence of small collocated data reads on an already-established connection. Here,

• initial connection overhead sees the higher latency (but is amortized over many data accesses),
• short synchronous reads mean TCP windows do not help,
• and the application can do as many interactions as desired, each costing delta-latency.

This scenario unfortunately is seen in the real world in some “skimming” analysis jobs, where individual events are read with minimal computing between reads.

3. Testing

3.1. Test setup

Identical diskserver1 from a single order have been installed both at Meyrin and Wigner, and have not yet been allocated to experiments. Of these, 10 EOS diskserver at each site have been added to two dedicated EOS spaces2 on the EOSPPS test instance, and another 5 each have been added to a “mixed” space. In each space, a series of 20 random test files for each size (1kB, 1MB, 1GB, 100GB) were created, each with 3 replicas – the intention of this replication is to remove adverse effects of a potential single “bad” source diskserver via consecutive test runs.

The machines have identical operating systems (SLC6), software versions, and configurations. To note that at the time of testing, several auxiliary services are not yet replicated at Wigner – in particular, all Kerberos KDCs are in Meyrin.

For the purpose of these test and where not otherwise stated, caching effects were ignored1.

---

1 AMD Opteron 6276, 2.3MHz, 32cores, 64GB, Mellanox MT26428 10GigE, 24x Hitachi HUA72303, 3x WDC WD2003FYYS-0
2 A “space” in EOS consists of a set of failure groups, each of which is a logical grouping of individual filesystems. Such “spaces” can be attached to directory trees, so that all files under a particular tree are located on disks in that space.
3.2. Harddisk local bandwidth – hdparm
In order to get an idea on obtainable single-stream performance in the absence of OS-level caching, a simple harddisk test was used to establish upper limits (ignoring disk zones, effects from concurrent access or OS-level tuning). The test was run on a subset of machines, not distinction between Meyrin and Wigner machines was made.

Our test machines consist of a CPU server with 3 locally-attached SATA disks, and a SAS enclosure with 24 disks. As Fig. 1 shows, most disks can deliver 155MB/s, and we can also infer that the local disks are somewhat slower (at 110MB/s). Most importantly, in our system single-stream performance above 160MB/s can only be attained via caching effects – so any delivered bandwidth at or above this speed will be considered to be “good enough”.

3.3. Latency test – ping
As can bee seen in Fig. 2, a simple “ping” cross-test between all machines confirmed that the network layout within each site indeed will not have a major impact on latency, and can be ignored for the purpose of this test (of course, other network parameters such as interface speeds or the ratio of up- versus downlinks - “blocking factor” - will affect overall performance, but these are identical between the two sites, and not relevant for single-stream tests). The twin peaks caused by the two different-length wide-area links (one provided by T-Systems, the other via DANTE) are just about distinguishable, but their RTTs are close enough to each other that they will be treated as a single “remote” link.

The bandwidth-delay product (BDP) of 28.75MB = 23ms*10Gb/s (the local interface being the limiting factor here) required tuning the Linux kernel in order for TCP windowing to work efficiently.

3.4. Throughput test – netperf/iperf
A similar cross-machine memory-to-memory network test via netperf and iperf3 discovered a sufficiently high error rate (CRC frame errors) that even generously broad confidence intervals (± 10%) could not be met for the “remote” tests – after each such error, it would take about 10sec for 1-sec bandwidth to reach pre-error levels. The issue is still under investigation and might be linked to the firmware on the switches rather than bad cabling. It affects both machines at Wigner and in Meyrin (but with much reduced impact for local access). However, since the worst obtained network bandwidth was still above the above disk speeds, we felt that this error did not necessarily preclude further tests.

3.5. Streaming transfers – xrdcp
The service-internal transfer speed and performance for bulk data transfers was assessed via a series of xrdcp runs for files of varying sizes, from the 3 EOS “spaces” set up before. The tests consisted in the client reading several files of a given size
from a given location (to /dev/null). Bandwidth was calculated from the runtime of the `xrdcp` command and file size (and so included full connection overheads).

As can be seen in Fig. 4, for small files the overhead of authentication, handshaking and redirection dominates the transfer speed. At these small sizes, the Meyrin - Meyrin transfers have the expected speed advantage over the Wigner - Wigner transfers, due to local Meyrin-only services. Also as expected, small “remote” transfers fare worst and are an order of magnitude lower than the Meyrin - Meyrin baseline.

However, as of 1GB file size, even in the presence of the above bandwidth-limiting network errors the disk speed is reached even for remote reads, and there no longer is a difference between local and remote transfer speeds. Under the above assumption on file sizes, the impact of small remote files on overall performance is assumed to not be significant.

3.6. ROOT macro event reading

In order to investigate the impact of the above worst-case scenario, a synthetic test based on ROOT macros reading files synchronously event-by-event has been run, in two variants:

- whole file is read sequentially (“100%” case) - Fig. 5
- 10%-percent random event subset is read in a loop (same amount of data as above) - Fig. 6

Both tests have been run in a number of scenarios:

- the data source was either in Wigner or Meyrin (client always was in Meyrin),
- local caching via ROOT’s TTreeCache (fixed 30MB cache) has been enabled and disabled,
- different ROOT versions have been used (with asynchronous prefetch on the TTreeCache enabled for the newest version)

In all cases, the metric was CPU efficiency (i.e. CPU time over WALL time).

The immediate impact of running a non-caching old client against a remote data source was rather impressive: in the worst case (old ROOT version, no cache) CPU efficiency dropped from 37% to 1.2%. However, the effects of turning on local caching were equally impressive, with a (rather moderate) 30MB TTreeCache performance bounced back to local levels. For more elaborate strategies like the asynchronous read-ahead on the local cache, the effect was application-dependent – whereas the sequential read and local random access benefited from asynchronous prefetch, the remote random access case suffered from the cache being swamped with later-unused data.

3.7. Upcoming tests – experiment

At the time of writing, ATLAS is preparing a series of tests via their HammerCloud infrastructure[3] that should investigate the impact of additional latency on production-like jobs. Since these will use more CPU respective to I/O, the effects of higher latency ought to be reduced. In parallel, a first series of batch nodes at Wigner has been put into production, and are being analysed – marked CPU efficiency differences will be spotted.

---

6 caching of course did also improve efficiency for purely local access

7 this counts as extra capacity beyond MoU allocation, so the experiments will still benefit from using this potentially inefficient computing capacity
4. Software-side improvements

4.1. ROOT client – TTreeCache
The results above demonstrate that local TTreeCache is beneficial even for LAN access, and indispensable whenever there is a risk of higher latencies. Until ROOT-6 (where it defaults to “on”) is widely deployed, experiments are encouraged to consider switching on such local caches. This is doubly true in the light of high-latency federated data access such as ATLAS FAX[4], CMS AAA or HTTP/WebDAV federations.

4.2. Geo-Location and service replication
On the EOS server side, a useful feature in the new EOS-0.3 release is the ability to “tag” data sources (via explicit configuration) and clients (via their IP network) with a location. These “geo-tags” are used in two cases:

- On writing, the algorithm tries to ensure that file replicas are spread apart. For the Meyrin/Wigner case, this makes it very likely that a replica is placed at either site (provided that sufficient diskspace is available on both)
- On reading, the EOS redirector with a high probability will direct a client from a suitably “tagged” IP network to the closest file replica.

Together these two mechanisms have been tried on a “mixed” EOS space (each file had at least one copy per site) under the above ROOT macro tests, and have largely eliminated the performance difference between local and remote reads. While the corresponding migration/rebalancing algorithms are still under development, increasing the Wigner capacity will be immediately beneficial - “hot” files tend to be “new” files, and “new” files will get spread to Wigner via the placement algorithm.

In the next phase, a local read-only namespace at Wigner will address the metadata latency penalty on read for Wigner clients. The read-only namespace will “bounce” any write operation to the read-write namespace, which is expected to stay at Meyrin. The effect of high latency between the two namespaces will have to be carefully investigated.

4.3. Conclusion
The capacity increase via the remote computer centre at Wigner comes at a price in terms of job efficiency. Efforts have been made within the EOS service to limit this impact. While the most types of job seem to be little affected, and the gain in capacity hopefully outweighs the efficiency loss for the rest, experiments and users are advised to tune their jobs to cope with additional latencies.

References

1: A J Peters and L Janyst, Exabyte Scale Storage at CERN, J. Phys.:Conf. Ser.,331-052015,2011
2: EOS - CERN disk storage system: http://eos.cern.ch/
3: P Andrade et al, Review of CERN Data Centre Infrastructure, J. Phys.:Conf. Ser.,396-042002,2012
4: D C van der Ster et al, HammerCloud: A Stress Testing System for Distributed Analysis, J. Phys.:Conf. Ser.,331-072036,2011
5: R W GARDNER JR et al, Data Federation Strategies for ATLAS Using XRootD, J. Phys.:Conf. Ser.,to-appear,2013