Storage Resource Reporting

Draft proposal v0.6, 07/09/2016
Alessandro Di Girolamo, Oliver Keeble

Introduction

This document summarises a proposal for storage resource reporting in WLCG, intended to enable experiment operations and WLCG storage accounting. The document proposes a small set of requirements which should be satisfied by WLCG storage systems.

The document also covers the WLCG accounting use case and discusses how the information provided could be used for accounting purposes. Storage accounting for WLCG is under review as part of a more general reconsideration of accounting and the accounting portal.

The GDB agreed to that the approach should be to finalise consumer side requirements (from the experiments and WLCG management, ie the first section of this document) and then take this to the storage providers. The stakeholders “just” need to say what they need, and how often.

Alessandro’s presentation at the GDB
https://docs.google.com/presentation/d/1hJ_bh8TzBdxLsi2WDIXaIZoUtySg_OHn2t4OtVMtg8w/edit#slide=id.g12418cc379_0_68

Proposal

The proposal has two parts. The first is related to resource reporting data delivered either through the relevant protocol or via a summary file, giving used and free space. The second part relates to “storage dumps” which are passed to the experiments periodically and present complete summaries of the system.

High level overview of the used and free space.

The interface and structure are not defined here, this is left to the storage providers. A storage system should implement resource reporting in at least one protocol. For a particular protocol,

the storage providers should agree on a single interface. NB – Alice request that all supported protocols provide this capability.

This information is intended to serve (at least) two use cases – determining the total capacity used and available to the experiment, and determining if there is sufficient space for files to be written.

We distinguish two entities which can both be understood by the word “quota”;

- a *capacity block*, when allocated, increases the total capacity available to the user. An SRM space token is a capacity block, also known as a quota node.
- a *restrictive quota*, when applied, limits total available space but does not add capacity

- Total used and total free space for the following
    - All distinct *capacity blocks*, available to the experiment
        - This will allow calculation of total volume provided by the site to the experiment. However, there is a “topology problem” here - the experiment must be able to identify these nodes in the namespace in order to sum them up.
    - Any entity on which a *restrictive quota* has been applied
        - This will allow clients to understand if they can write
- Used space under a directory should not take into consideration files stored in different *capacity blocks*. In other words, it should be “du -x” rather than “du”. This will allow summing of individual entries to give a full view of capacity.
- Relevant numbers should be available for tape.
    - The disk part can follow guidelines for disk-only systems
    - The tape part could publish the json summary, for used space.
        - Aggregations would not be based on path but some other meaningful grouping, e.g. tape families, or could simply be a single number per V.O.
        - Free space – we propose that this is not required
- Query frequency - can be up to half an hour (not Hz)
- Accuracy
    - volume - order of tens of GB (i.e. experiments are not picky on super precisions, Storage providers should comment on what’s doable with a limited amount of complication).
    - time – a freshness of tens of minutes or so is acceptable

Over-commitment, replication & maintenance

Over-committing storage can happen in a number of ways

- In the current SRM system, space tokens (eg 5PB & 5PB) can be defined on a disk pool whose capacity is, or becomes, less that 10PB
- *Restrictive quotas* can be imposed, whose sum exceeds available capacity.

Replication can be exploited by a system to improve resiliency and increase performance for some operations.

We propose that over-commitment and replication be viewed as internal to the system and not be exposed. This simplifies the reporting system considerably.

Parts of the system can be deployed but unavailable. What should used/free numbers reflect?

Clients

The relevant numbers will be made available through the gfal2 interface to extended attributes (gfal2_getxatt etc.)

json summary

Atlas have proposed a json summary of the storage system be made available at a well known location within the experiment namespace. It would be accessed at a well-known location within the namespace - <basePath>/lhcb/spaceSummary.json.

The json file has two advantages

- it solves the “topology problem” of knowing which parts of the namespace to query in order to assemble a complete view of capacity
- it can be cached to allow high query rates

This json file would show a single entry per *capacity block* (not mentioning *restrictive quotas*), such that its contents can be summed to give the full capacity allocated to and used by the experiment. It can be thought of as the complementary “df” analogue for the “du -x” discussed above.

The appendix includes a possible schema and example json files for a number of scenarios.

More detailed storage information

Full storage dump enumerating each file. The aim is to allow a single utility per storage system which will work for all interested experiments. The following information represents the union of the attributes requested by Atlas, CMS & LHCb.

- path
- size
- accesstime
- ctime
- checksum type & value

To be provided on weekly timescale when not possible with higher frequency (e.g. EOS find allows, in a couple of hours, of having the full dump. CMS run it daily).

Usage for WLCG Accounting

Only the summary information described above is required for accounting. The workflows of the experiments related to space accounting won't change, but agreeing on the format, structure and content of the information exposed will enable the possibility to setup a WLCG collector in parallel to the experiments workflows to collect the Storage Resources accounting information.

For transport to the accounting aggregator (currently APEL), there are a couple of options

- Storage systems emit accounting records and send them to APEL
    - There is a varying amount of existing support for StAR accounting records, and this standard is under review by EGI who will provide the accounting portal
    - There is a newer standard, OGF UR V2, derived from StAR, to which APEL expects to migrate
- A "pull" model where APEL or an intermediate service gets the data from the endpoints and creates the relevant records. This would avoid any modifications to the storage systems purely for the accounting use case.

Requirements Overview

The following table summarises which information is required, interesting or not required for each stakeholder.

	Summary info	Detail storage dump
Alice	Via all supported protocols	Not required
Atlas	Via at least one protocol or via json	Path. Optional - size, atime. Once per month
CMS	Via at least one protocol, or json if protocol is not possible	Path, size, checksum
LHCb	Via at least one protocol protocol, json (?? tbc)	Path, size, ctime. Once per week
WLCG	Either via at least one protocol or json	Not required

Appendix A

Example json objects

```
[
        {"capacity_id": "capacityBlock1",
        "status":"online/offline",
        "status_message": "The report can not be created because ...",
        "list_of_paths":
["/castor/ads.rl.ac.uk/prod/atlas/stripInput/atlasdatadisk/","/castor/
ads.rl.ac.uk/prod/atlas/stripInput/atlassdfsdfdsfsd/",...],
        "total_space": 5000000000,
        "used_space": 2000000000,
        "num_files": 123456,
        "time_stamp": 1447936989},

        {"capacity_id": "capacityBlock2",
        ...
        }
]


[
```

```
        {"capacity_id": "ATLASDATADISK",
        "status":"online/offline",
        "status_message": "The report can not be created because ...",
        "list_of_paths":
["/castor/ads.rl.ac.uk/prod/atlas/stripInput/atlasdatadisk/"],
        "total_space": 5000000000,
        "used_space": 2000000000,
        "num_files": 123456,
        "time_stamp":, 1447936989},

        {"capacity_id": "ATLASSCRATCHDISK",
        ...
        },

        {"capacity_id": "ATLASLOCALGROUPDISK",
        ...
        }
]
```

Appendix B

Here we describe the numbers expected from an example system. The scenario is deliberately slightly contrived, in order to illustrate the relevant points.

Points to note;

- The diagram shows the expected resource reporting numbers, and where they should be reported (ie by the relevant protocol frontend, or through the json summary).
- The diagram shows a hierarchy with two paths of interest, /dir1/dir2a/dir3 and /dir1/dir2b.
- Each directory has 1TB of data in it
- /dir1/dir2b has a separate *capacity block* allocation, perhaps a spacetoken.
- Overcommitment of resources is not exposed.
- /dir1a and /dir1a/dir2b would be identified as the paths to query for total capacity numbers (and thus appear in the json file).

Resources reported