CLUSTER DISTRIBUTED DYNAMIC SOTRAGE

A. Forti, A. McNab University of Manchester, Manchester UK

Abstract

The HEP department of the University of Manchester has purchased a 1000 nodes cluster. The cluster will be accessible to various VOs through EGEE/LCG grid middleware. One of the interesting aspects of the equipment bought is that each node has 2x250 GB disks leading to a total of approximately 400 TB of usable disk space. The space is intended to be managed using dcache and its resilience features. The following describes different dcache configurations and disks and network layout adopted to exploit this space. Different configurations can be used to target different use cases. An alternative method based on http technology will be also described.

BACKGROUND

The University of Manchester has bought a 1000 nodes, 2000 processors cluster and ~400 TB disk space available across the nodes. The cluster is mainly dedicated to High Energy Physics and a small fraction for other sciences. The cluster is grid enabled, there are 14 active VOs on it and each of them has different storage requirements depending on their computing model. The decision of going for commodity computing both for cpu and storage was driven by two reasons.

The first reason was the most important physics experiments have tiered structure computing models. The main difference between Tier0/Tier1s and big Tier2s is the storage space and the quality of service for it. Tier2s will have an importance mostly for cpu resources they cannot afford to supply a safe, i.e. very expensive, storage and the manpower to support it. However they still need space for data to be analysed by the end users and to store the output of production data like simulation or reconstruction.

The second reason was that the cluster would have been on the grid. The most important strategy to keep data safe is resilience. When the environment was not distributed and each site could rely on ly on local resources to protest their datas it was obvious that local backups, more expensive storage or various replication technologies like raid arrays were necessary. On the grid everything is connected and this need is not anymore so compulsory. Data can be replicated across sites, other Tier2s for example. Resilience is moved a level up.

So in Manchester the decision was to buy as many cpus as possible and the biggest disks it was possible for the best price/size ratio at the time. There is no tape storage behind and no backups are foreseen. The clluster internal connectivity is 1 Gb/s and is connected to the external world by the 2.5 Gb/s university production network. It will be replaced by a 10 Gb/s dedicated link. High rate data transfers from and to Tier1 are possible. 350 Mb/s was achieved on the shared university production network.

The aim is to use the disk space on the nodes as a cache. Since the nodes will serve and process data at the same time it is important not to overload them with requests for specific data. Replicas on the cluster itself are therefore foreseen for load balance rather than for safety.

SOFTWARE

Today there is a choice of different storage management software that can be installed. To name the main ones available in the grid environment: dcache [1], DPM [2] and xrootd [3]. A solution based on http technology is also being developed in Manchester. At the eyes of the user there shouldn t be any difference as two most important characteristics of these softwares are that they offer a common name space and eventually an srm interface. However at the eyes of the administrator the stoarage system has to be scalable and also offer some automatic data management features. For this and for historical reasons dcache has been the software of choice.

Dcache

The main dcache characteristics that were looked at:

- Combines several hundreds pool nodes under a single name space.
- Support multiple internal and external copies of a single fuile system entry point
- Performs automatic pool to pool copies of datasets to flatten data access
- It has fine grained pool selection (experiment, read write, internal external, priority)
- Cached data are removed only if space is running short, while precious data are kept even if the system is hungry for space. Which means pools can be configured for different needs.
- It can maintain a minimum of m to a maximum of n copies on one system. i.e if a pool with one copy of the data goes down the number of copies is

maintain constant. This obviously doesn't work if there is only one copy of the data on the system. However is very useful for maintainance.

- It supports various access protocols
- Since it was designed for tape and disk it doesn't really matter what hardware is underneath so different type of hardware with different levels of redundancy can coexist in the same dcache instance and be flagged differently.

As an aside it has already been used to manage storage on batch nodes by Fermilab and BNL Tier1s.

Configuration

The system will have front end nodes with two interfaces. One of the network interfaces will be dedicated to external transfers and will have an active gridftp door on it (Figure 1). Since all the nodes have more than a cpu the griftp door can also have a dedicated cpu. Internal nodes will only be accessible from other nodes.

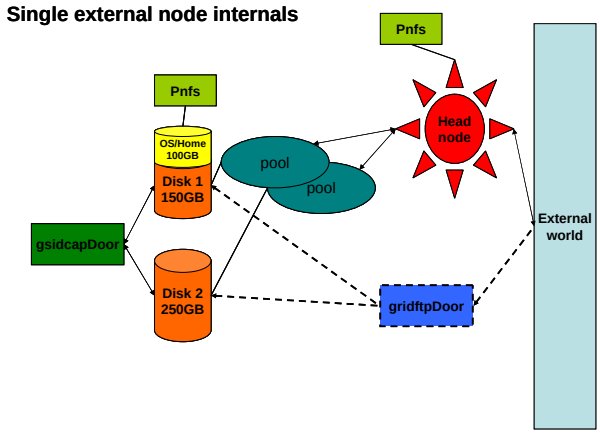

Figure 1: Front end nodes configuration

The aim is to optimize the space and minimize the impact of maintenance. On each node there are (Figure 2):

- 2 dcache partitions 1 on each disk
- 2 pools, 1 for each partition
- /pnfs mounted
- Gsidcapdoor active

No raid arrays will be created it spoils the independence of the discs. In this way

- Disk 2 can be assigned to a particular VO
- Disk 1 can be used for resilience.
- Disk 1 can be wiped out during installation if needed
- Disk 2 can be left untouched.

Data will have at least 2 copies.

Internal nodes access.

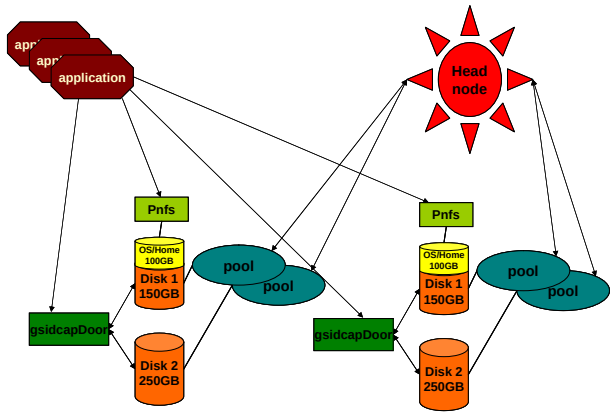

Figure2 : Internal nodes configuration and application access

Applications

Applications will access data on other nodes using either gsidcap doors enabled on all the nodes or /pnfs and LD_PRELOAD environment variable. Root applications can benefit from the API distributed with root toolkit.

Other solutions

As part of the GridSite project, we have also developed an HTTP(S) based file storage solution. This makes use of the GridSite security extensions to the Apache web server for file access, and an additional layer of file location is provided by SiteCast, which uses UDP multicast to find replicas of a given file within the site.

GridSite provide fine grained file access control in terms of the X.509 digital certificates and VOMS attribute certificates possessed by users of the LCG and gLite middleware. In addition, it implements the write methods PUT, DELETE and MOVE from the HTTP and WebDAV IETF standards. Combined with directory listings, and the HTTP HEAD method to query a file's status, this is sufficient to implement the functionality of a traditional FTP or GridFTP file server, but with fine grained access control, based on groups and roles with a VO, rather than coarse grained access based on the local Unix account permissions.

This allows individual worker nodes in a farm to give access to files stored on their disks, but the problem remains how to located the files. Rather than implement a heavy weight solution involving a central database machine, we have chosen to add a multicast responder to GridSite, to enable the web servers themselves to respond to queries. Since GridSite creates new files in an atomic way, by creating a temporary file which is renamed when

writing has completed, the information about whether a file exists and what size it is, is available directly from the file system. This obviates the need to maintain a database, and means that if worker nodes go offline, their files are immediately removed from the virtual database. This SiteCast system uses the IETF's Hypertext Cache Protocol (HTCP) to communicate queries and responses.

We have also added support for SiteCast queries to the htcp suite of command line tools supplied with GridSite, which allows copying of files from virtual SiteCast locations within a farm to a local disk, and we have begun design work on a system to give POSIX filesystem access to files stored on GridSite and located by SiteCast.

REFERENCES

[1] http://www.dcache.org
[2] http://www.grif.fr/article.php3?id_article=16
[3] http://xrootd.slac.stanford.edu
[4] http://www.gridsite.org