dCache, Q&A

Patrick Fuhrmann[1] for the dCache team

Deutsches Elektronen Synchrotron
Notkestrasse 85, 22607 Hamburg

Abstract. This writeup is the result of a set of questions sent to Storage Element providers, in the context the storage day at HEPiX 2006, in order to allow the audience to compare Storage Element functionality and more. With that, only selected topics are touched. For a detailed discussion of the dCache project please checkout *dCache, the Book.*

1 Functionality

1.1 Policy Management

Disk Pool Management in general For each dCache I/O request, regardless of the dataflow direction or whether an HSM is involved or not, dCache performs a two step process to determine the most appropriate pool for that particular request. In the first step, the central decision module applies administrator defined rules resulting in a group of pools allowed to be used for that request. The decision is based on the IP number of the data client, the data flow direction and the subdirectory tree of the requested file. With the next release dCache will take the data transfer protocol type into account as well. This step basically allows different user communities to provide their own storage hardware which they don't need to share with other groups. Moreover, different types of storage hardware may be assigned to different classes of user data. As a result of the first decision step a file might be transferred to a different pool before it can be delivered to the client in order to enforce the access rules. As a second step, the resulting set of pools is checked for availability and a cost mechanism is used to pick the final pool. The cost metric is build upon the number of active transfers and the access time of the least recently used dataset per preselected pool. At this point a pool to pool copy might be initiated if overall cost can be minimized.

Disk Pool Management related to HSM interactions dCache supports external copies of its file repository. Those copies usually reside within an attached tertiary storage system. For getting data out of those external systems, all rules described above apply as well. If more than one HSM system is involved, pools can be selected based on the decision of the administrator to which HSM certain files should finally go or should be restored from.

Files requested, but not cached on one of the dCache pools are immediately requested from the connected HSM systems. Its up to the HSM to delay and

group those requests for optimization purposes. This is different for files scheduled to be transferred from a dCache pool into an HSM. Those files are grouped by dCache and flushed into the tertiary storage system if certain thresholds are hit. The grouping is user defined and mainly based on the subdirectory tree structure. The decision, when to flush a bunch of files into the HSM is autonomously done by the related pools. It turned out that for certain purposes this approach is not yet sufficient. e.g.: i) Some tertiary storage systems come up with constrains enforcing central management. ii) For high performance applications it would be desirable to disable HSM flushing as long as data is flowing into a pool and on the other hand suppress incoming data while the pool is being flushed into the HSM. iii) Similar to collecting data before sending it to the HSM one could think of collecting HSM read requests before forwarding them to the tape system.

In order to cope with those ideas, we are introducing a central HSM flush module which for convenience will come with sufficient functionality to cover the most common cases. Moreover, there will be a clear programming interface to allow site administrators to write there own drivers optimized for the HSM in use. This module is currently in beta testing.

In case of an improved HSM restore procedure we are evaluating the possibility to let a central but external module decide when a certain restore request should be forwarded to the connected HSM.

2 Priorities

On each data pool, dCache handles separated data transfer queues per access point or door. In the easiest case, a door, or set of doors is equivalent to a protocol type (e.g. dCap , gsiftp). Because each pool can be individually configured on how many active transfers it allows for each of those protocols, a 'priority like' mechanism is in place per protocol. Furthermore, because pools are selected based on the user community, resp. subdirectory tree, protocol priorities may be different for different user groups. Moreover, the administrator may decide to have multiple dCap queues per pool and the dCap protocol provides mechanism to choose among those queues, so that e.g. for the dccp application less active transfers are allowed than for a user application linked against the dCap library.

3 Access Protocols

dCache currently fully supports the SRM version I and most parts of Version 2.1 except for the explicit space reservation functionality. gsiFtp is supported except for the upcoming "improved extended block mode (MODE X)". Theoretically http is supported but has never been used so that experience is not available. dCache supports its native dCap protocol for posix like access. We provide a C library to be linked against the user application or a linux and solaris preload library to support applications which can't be recompiled. The xrootd protocol

has been implemented and will undergo serious testing mid of April. An API is available to add other protocols as long as they support data stream redirection.

4 Authentication / Authorization

Currently dCache only supports access permission based on the standard unix permission semantics. Chimera, the new dCache file name space provider, expected not before mid of 2006, will support ACLs though it's not perfectly agreed to which extend.

With the next release, dCache will have gPLAZMA integrated, offering to have multiple pluggable modules handling the extended X.509 proxy certificates, legacy static files and possibly allowing to directly talk to the VOMS service.

5 Support for transactions

Based on an agreement within the dCache collaboration, interrupted ftp transfers will result in the destination file to be removed while interrupted dCap transfer will keep the file entry and will store as many bytes as it could get during the transfer. Only zero byte files are removed from dCache if the 'close' operation hasn't been issued.

6 Access to namespace service (by the client)

Beside the standard protocols like http and ftp which define how to address a file, dCap allows to either address files directly by their name in case the pnfs filesystem is mounted or by providing an appropriate URL style address. Namespace operations like ls, rm, mv, etc may be done directly in the mounted pnfs filesystem using the corresponding OS tools or by using the dCap API with URL style addressing. Furthermore, having the dCap library preloaded, OS native tools may be used for namespace operations together with the URL like syntax without having pnfs mounted

7 Scalability

With dCache installations exceeding 300 TBytes located on some hundreds of pool nodes, we didn't hit the maximum number of storage nodes resp. the maximum total amount of space the dCache yet. In those systems we observe that the overall transfer rates still linearly increases with the number of pool nodes. Theoretically the expected limits are far beyond several petabytes resp. several thousands of pool nodes. The only limitation we ran into yet is the number of file 'open' operations per time unit. The actual number depends on the performance of the name service host but is always below 50 op/second. Chimera, the upcoming name service, has already shown to improve this by at least a factor

of 10. We expect this service to become part of the dCache mid to end of the year for sites willing to do serious real world testing. The time it takes to open a file in dCache is in the order of several 100 ms up to a full second. This is an intrinsic delay which results from the fact that quite some calculation has to done to determine the best pool. This number is not directly related to the number of 'open' operations per second because the Pool Management modules are highly multi threaded and can perform those decision processes independently. The total number of files in the current name space system (pnfs) may exceed several millions without penalties if the database structure is setup accordingly. Chimera doesn't need tuning to scale up to several 10 million of files. Beside, while the current name space provider is bound to a single machine, Chimera can be spread over a set of hosts. To our current experience the amount of data delivered by a single dCache pool is limited by the transfer speed of the network interface only.

8 Flexibility

dCache defines a clear interface to backend tertiary storage systems. Simple shell scripts may be used to flush and retrieve data into resp. out of such a system. Though the interface itself is simple the underlaying mechanism may well be used to optimize the work of tertiary storage. (See above)

Although we are preparing for Chimera which intrinsically supports to be run on distributed machines, we already support splitting up the currently used file name provider instances for scalability.

Except for the name space provider and the client library, dCache is totally written in java. So any OS supporting the JVM 1.4 and higher may be used to run the dCache server software. The file system provider will follow mid of the year and will, at that point become OS independent as well.

9 IT Security

dCache data is owned by the 'root' user on the various dCache pools. So anybody having 'root' access to dCache service machines can produce arbitrary damage. The dCache nfs2 interface, if enabled, is exactly as secure as the nfs2 protocol itself, so essentially not at all. If used, it's strongly recommended to have firewalls in place to at least protect from illegal remote access. If local dCap file system access it not required, nfs exports can be restricted to localhost of the headnode and, in case of a connected HSM, to the write pool hosts.

Both, FTP and the url based dCap access may be GSI protected using X509 certificates for authentication.

Buffer overflow attacks are extremely unlikely with java applications.

The name space provider can be optionally backed by the postgres database which provides mechanisms to allow the databases to be online replicated to other hosts. At any time, the replicated database is an exact copy of the original one which allows fast swapping in case of hardware dropouts.

10 Configuration management

Though dCache is coming with a reasonable setting for a mid range installation, decent customization needs to be done when growing above a certain threshold. A better understanding of the system is required in case of an HSM connection, dual homed head or pool nodes and more sophisticated network configurations. Most of this is described in the "Cookbook" section of "dCache, the Book".

11 System Operation

A dCache instance itself presumably needs about 1 FTE to be build up and less than .5 FTE to be operated long term. This is relatively independent of the size of the installation. For dCache systems connected to tertiary storage this is slightly higher though most of the effort goes into the HSM part. This estimation assumes that the disk and server hardware is operated by some system group. The work going into that certainly depends on the number of installed servers and the quality of the hardware and professionalism of operation.

dCache provides a command line interface to the full command set allowing scripting and graphical interface for convenience.

12 Support

One full FTE is allocated for organizing dCache support, packaging and documentation. Both, DESY and FERMI have agreed on a continues dCache support for a predicable future and not only because both of them highly depend on that software. Moreover, with the growing user community, more labs offer help for support and development. About 6 FTEs are working full time in dCache design and development while more than 10 are involved in special tasks. Base on this, other projects have been spun-off integrating dCache into various frameworks. Altogether dCache knowledge is spreading rapidly.

13 Packaging, product distribution and documentation

User interaction with dCache is funneled through the www.dCache.org web pages. We offer a download area for the most recent 'rpm's', pointers to talks and papers about dCache and growing documentation in "dCache, the Book". Moreover support@dCache.org provides help in case of trouble and in case of unusual use cases. With the user-forum@dCache.org mailing list, dCache users have the opportunity to exchange information and experience and may provide help to each other.

14 License Model

dCache is free of charge for non-profit organizations. Regular support is provided on best-effort-bases. Prioritized support is available but needs to be compensated.