dCache, Selected Topics for the LCG MB.

Patrick Fuhrmann, Aug 4, 2009

Content

- Chimera and the Chimera-to-Pnfs migration.
- dCache, Golden Release for the initial LHC run.

Chimera and the Pnfs to Chimera migration

Introduction

One of the central components within the dCache storage element is its file system engine. The engine basically translates a hierarchical name-space, humans are used to deal with, into a flat i-node space. Essentially all dCache operations require one or more requests to be sent into the name space system. Consequently the performance of that component is crucial for the performance of dCache itself. Originally the dCache file system functionality was based on Pnfs. Pnfs is a rather old technology which wasn't designed to work with dCache. Its main deficiencies are

- Pnfs can only be accessed through its NFS2/3 interface. Even the core dCache system has to use this channel which is causing an unnecessary performance bottleneck.
- Pnfs database accesses are protected by a single lock which prohibits the software from scaling performance with the number of CPUs or cores.
- As the data records in the Pnfs databases are represented as binary BLOBs, no database queries (e.g. SQL) can be performed to allow system maintenance or monitoring.

As it became obvious, that Pnfs wouldn't be able to scale into the requested Petabyte storage range, dCache.org developed a successor to Pnfs (Chimera), based on modern database technologies. Chimera is officially available as a replacement for Pnfs since November 2008. Other than Pnfs, Chimera is just a database table layout and a library, simulating a file system. As there is no Chimera daemon and as database locks are avoided wherever possible, Chimera provides the ability to scale with the number of CPUs/cores and with the performance of the underlying database technology. Most important is that Chimera has been designed to work with dCache and that future performance enhancements and features in dCache certainly will be based on Chimera as the underlying file system engine. An example for a feature only available in Chimera is the introduction of Access Control Lists with dCache 1.9.3. Although ACLs are support by both, the pnfs and the chimera file system backend, the inheritance of ACLs is only available with Chimera. ACL inheritance is especially important using automatic subdirectory generation as part of the SRM protocol. Without ACL inheritance, the ACL's of auto-created directories will have to be set manually after their creation.

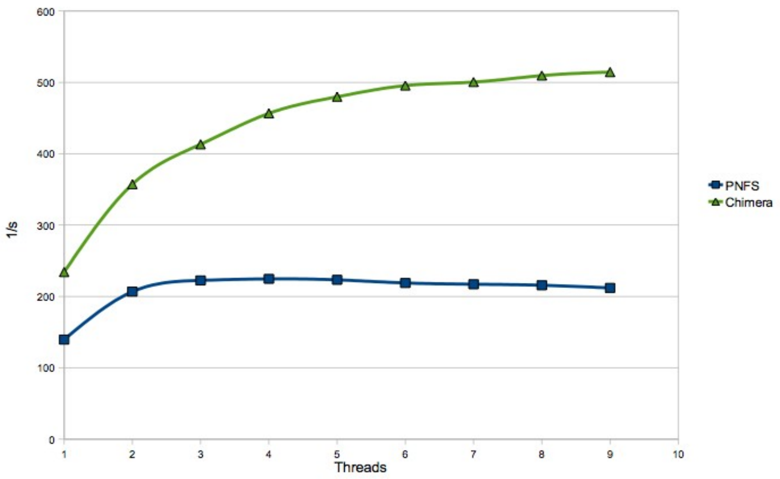
Advantages of running Chimera

As already described above, Chimera is based on modern database technologies and only consists of libraries directly accessing a standard database system.

Scalability : As no intermediate servers are involved, Chimera nicely scales with the number of available CPU/cores and with the performance of the database back-end. The graph below sketches the rate of fetching realistic information out of Chimera compared to Pnfs. While Pnfs doesn't scale at all with the number of threads. Chimera scales up to the point were the number of threads approaches the number of available CPUs.

Maintainability : Within the Chimera databases, all entities are stores as the type they represent. This allows system administrators to use the standard SQL language to query Chimera for all kind of information which otherwise would be very difficult to gather. This might be the amount of data a single user or group is using within dCache or the amount of data stored on Tape for a particular storage group.

Future : Further development of dCache already assumes Chimera to be the underlying file system engine. In consequence, feature and performance improvements are essentially based on Chimera, certainly because we can not afford to work on two systems but mainly because of the fact that Pnfs is an outdated technology not allowing further improvements.



File system access rate by number of threads.

Migration from Pnfs to Chimera

dCache.org is providing tools to convert existing Pnfs instances into Chimera. The conversion consists of three step. Dumping Pnfs into the SQL database language. The injection of the intermediate SQL data into the new database and the verification of the correctness of the result by check-summing the Pnfs and Chimera name-space entries and comparing the result. All three processes may run concurrently for different Pnfs databases of a single instance. The converion of the 8 Million files at the NDGF Tier I took:

Dumping Pnfs into SQL	11 Hours
Injection SQL into DB	3.5 Hours
Run md5 verification	11 Hours

There is room for improvement by using appropriate hardware and smart concurrency.

The Pnfs dump tool is performing a consistency check of the Pnfs file system while doing the conversion. As the conversion is non intrusive and can be done independently of the production system, dry runs can be performed until all inconsistencies are resolved or fixed so that the the actual conversion can be well planned. We strongly discourage to run a Pnfs to Chimera conversion without having the entire procedure tested at least once.

The dCache.org web-pages provide sufficient information on the details of the conversion as well as material from a workshop on that topic held in Aachen in April '09.

dCache.org doesn't plan to provide tools to convert back from Chimera to Pnfs, we would instead treat any possible problem with Chimera with highest priority. However, the source code of Pnfs as well as Chimera would be made available if people would like to work on a back-conversion tools. The source code of Chimera is already available on our web pages to encourage people to review the system.

Risks Assessment

The conversion : The conversion procedure can be arbitrarily repeated on real Pnfs data without any interference with the active production system. This allows to eliminate possible inconsistency in the existing Pnfs file system and a very realistic estimate on the overall conversion time. Inconsistencies may have sneaked into Pnfs over time on events like server crashes e.t.c. They are reported by the conversion tool and can either be fixed by the system administrator or by dCache.org. Although it is technically not necessary to shutdown Pnfs during the final conversion procedure, this would be advised to avoid discrepancies between the old and the new system. In the unlikely event of a failing final conversion, it is possible to switch back to Pnfs as long as no files have been created in Chimera.

Risk of running Chimera : Chimera systems are in production since November 2008. The first larger site, the NDGF Tier I, migrated to chimera end of March this year. Nearly all German Tier II's, as well as sites in Italy, US and the UK have already migrated or are planing to do so in the near future. Concerning proper risk assessment, this is a decade of solid experience with Pnfs compared to moderate experience running Chimera but including STEP 09 which was supposed to be a realistic simulation of an LHC run.

However, non of the sites running chimera have been reporting issues which were related to Chimera itself. Moreover, as the Chimera database layout is public and Chimera is using standard database technologies, no in-depth knowledge of dCache or Chimera is needed to investigate in issues which may arise when running Chimera. This is completely different with Pnfs. A Pnfs inconsistency can only be fixed by a maximum of two people at dCache.org.
Another factor on the negative side is certainly that we don't provide tools for converting Chimera back to Pnfs. Beside the fact that that would have been a technological challenge, it would have taken away man power from stabilising Chimera and from simplifying the Chimera to Pnfs conversion procedure. As some kind of compensation, dCache.org is treating possible future Chimera issues with highest priority assuming that those theoretical issues can be fixed faster than a backward conversion would take.
Independent of the use of either Chimera or Pnfs it is inevitable to have an system administrator on site which is reasonably familiar with the database back-end in use which might be Postgres or Oracle.
Risk of running Pnfs : The technology, Pnfs is build upon, doesn't allow any significant improvements any more. If during the LHC run period, Pnfs fails to cope with the required data access pattern, there would be no other choice than converting to Chimera. Upgrading hardware is of course always possible but the benefit would be moderate as Pnfs can not really make us of more CPU/cores. The probability of Pnfs not performing properly is difficult to estimate and highly depends on the data access pattern and the setup of the site. The performance of Pnfs/dCache in the past and especially during STEP 09 can be taken as a hint. However if the *Pending Queue* within the *PnfsManager* tends to be non-empty for significant time intervals, converting to Chimera is inevitable.
As mentioned in the paragraph above, serious inconsistencies in Pnfs can only be fixed by a small number of people at dCache.org and may cause very long downtimes. The problem in Pnfs would have to be fixed before a migration to Chimera would be possible.
Inconveniences running Pnfs : More and more new functionality, build into dCache, may rely on the existence of the Chimera name space as they can not be realised with the Pnfs technology. Those might be new features as well as performance improvements. One example is the inheritance of ACLs. ACLs have been introduced with dCache 1.9.3 and can be used with Pnfs and Chimera. However, when creating a subdirectory, the ACLs of the parent are only inherited if Chimera is providing the name space. ACL inheritance is important with the auto directory creation offered by the SRM protocol.
Maintaining, accounting and monitoring of the file system is significantly easier with Chimera as the back-end database is used in a standard fashion. Queries like getting the amount of data a user occupies on disk or on tape can only be done with Chimera.
Last but now least, the NFS 4.1 server interface, which is available starting 1.9.3, strictly requires Chimera.

The dCache release policy and the Golden Release.

dCache.org is introducing an important change in its release policy. As many other software projects, we are moving away from feature driven releases towards time based releases. This essentially means that we can tell more precisely when the next release will be available, but no longer exactly which features will be made available within that version. The advantages for the dCache project mainly are that

- we can ease the synchronisation process between dCache and our distributors, that
- sites can plan well ahead for system upgrades and that
- releases are not infinitely postponed by waiting for promised features, which may be delayed for whatever reason.

Another common release policy is the introduction of *Golden Releases.* As you may already guess, this Golden Release is going to be supported throughout the entire first LHC run-period and, as usual, no new features will be added to this release to ensure stability. However, we will of course continue to publish new releases with new features. It's up to the site to decide either to stay with the Golden Release or to follow up on the feature branch.
The Golden Release candidate is dCache 1.9.5 and the publishing date is scheduled for end of September resp. beginning of October.
With the release of 1.9.5 the 1.9.2 will no longer supported as 1.9.3 will no longer support after 1.9.6 is out, etc.

The most prominent features of the Golden Release will be ACLs, Tape Protection and an improved algorithm for triggering pool to pool transfers.