

LDR Meeting Summary and LDR Plans for S4-S5

Attendance and Agenda

A one day meeting to discuss LDR and plan for S4-S5 was held April 23 at MIT. Attending the meeting were Keith Bayer (MIT), Kevin Flasch (UWM), Mike Foster (PSU), Ben Johnson (LHO), Scott Koranda (UWM), Brian Moe (UWM), Hari Pulapaka (CIT), and Murali Ramsunder (PSU).

The agenda for the meeting and copies of the presentation made are available at <http://www.lsc-group.phys.uwm.edu/lscdatagrid/doc/ldrmeeting.html>.

LDR Development Plans

The group listed and discussed in some detail a number of enhancements planned and needed for LDR. Below we detail each of the enhancements or features and give an estimate of the difficulty required to implement the feature along with a priority ranking.

Difficulty Index

The difficulty index is a number from 1 (one) to 10 (ten) with 1 being very easy and requiring no significant changes to existing LDR architecture/code and very little time and 10 being difficult and requiring substantial changes or additions to existing LDR code and a significant time investment.

Priority Ranking

The priority ranking is one of the letters **A**, **B**, or **C**. **A** should be interpreted as absolutely required for S4-S5, **B** as not required for S4-S5 but still hopeful for S4-S5 or soon after the beginning of S5, and **C** as not likely to happen before the completion of S5.

A Note about LDR's Core Mission and Requests for Enhancements and Features

The group agreed that LDR's core mission needs to be replicating LIGO and GEO data sets from source sites to analysis sites quickly and robustly. The highest priority has to be making LDR as robust and fault tolerant as possible so as to require as little intervention from administrators as is possible, and no functionality or enhancement should be contemplated if doing so detracts from this goal.

Opening Up LDR?

Before detailing the development plans for LDR we note that there exists pressure to "open LDR" so that LDR is not only used by administrators to replicate bulk data such as interferometer data and SFTs, but also to allow any LSC user to publish any files into LDR and have them replicated (if so desired) to other LSC sites.

This use of LDR was discussed at the meeting. There is much concern that the requirements of replicating user data and bulk "collaboration" data are different enough such that a second product similar to LDR but specialized for users should be developed

and deployed. At the same time it is recognized that the LSC probably doesn't have the FTEs available to develop and deploy such a product and it might have to rely on LDR for this task.

More discussion is necessary, though Scott and Brian have agreed to start to explore what changes to LDR would be necessary in order to allow any user to publish into LDR.

Detailed Development Plans for LDR

New metadata schema and metadata propagation: difficulty 8; priority A

Scott Koranda explained during his talk that the current implementation of the metadata table schema and the propagation method was purposely naïve and is now leading to performance issues for LDR (by putting undue stress on the MySQL server) and a lack of scaling as more files are published into LDR.

Brian Moe has designed a new metadata table schema and propagation architecture. The metadata table schema is based on a schema used by the Globus group at ISI in the Metadata Catalog Service (MCS) product being developed. LDR cannot leverage MCS at this time because MCS does not have a model for propagation of the metadata information.

Details of the new schema and the client/server propagation method are in Brian's talk available at the meeting web page.

The difficulty of this task is high since it requires a fundamental change to the LDR architecture. This change, however, is absolutely necessary for S4 and will actually be part of the next LDR release that is planned to happen soon.

Speeding up the transfer of small files: difficulty 6; priority A

Due to the fundamental limitations of TCP transferring small (less than 100 MB) files over the WANs we use is much less efficient than transferring large files. The difference in transfer rates can easily be a factor of ten.

We plan to follow the lead of a group at IBM and develop a customized GridFTP-enabled server which would be able to respond to a command from a customized client to tar a number of files together and transfer them as a single large file. We cannot directly leverage the work the IBM group has done since they used the Java COG from Globus but we have a number of ways that we can proceed including using the new GridFTP server class in the Python-based pyGlobus package and a completely new GridFTP server code base coming from the Globus group that is not based on the wuftp code and which has been designed to be very extensible. The Cactus group has used a beta version of this new server code base to create a server that can deliver only "interesting" parts of files as requested by the client.

The difficulty of this task is moderate since we will have a fair amount of new development to do, but not too high since we can leverage to some degree the work done by other groups as noted above. This task is required for S4-S5 since most sites are now interested primarily in the level 3 RDS of LIGO data and these are fairly small files.

Authorization for data access at the LDR level: difficulty 8; priority C

At this time authentication within LDR is done with Globus GSI but authorization is handled only at the UNIX filesystem level. A 'datarobot' agent from a site such as UWM uses a X.509 certificate and GSI proxy to authenticate to a GridFTP server and then the certificate is mapped to a local UNIX account such as 'uwmrobot'. If the 'uwmrobot' account has the correct UNIX permissions to access data then access is authorized.

Ideally authorization to access various data sets should also be handled within the GSI security infrastructure. A 'datarobot' presenting a certificate from UWM for example could be given access to only certain data sets and not to others without any dependence on the details of the underlying filesystem.

A large amount of research is being done within the Grid community to make authorization via GSI a reality and there are a number of products available for us to test. Integrating this into the LDR architecture and code base, however, would require a large effort. At this time we do not see this as necessary for S4-S5.

Smooth upgrades with migration of state in MySQL; difficulty 3; priority A

Upgrading to the next release of LDR or incorporating new features into a deployed LDR should be easier and should not require tedious work with the MySQL server. We plan to make sure that the current state at a site can be preserved and that any migration to a new schema for example is handled automatically using scripts. This work is not very difficult and will be in place for all future upgrades/releases.

Choice during install to use existing MySQL server: difficulty 2; priority A

When installing LDR one ought to be able to use an existing MySQL server rather than having a new one installed. This is not difficult and will be done as part of the next release/upgrade.

"Green/Red light at-a-glance" LDR status web page: difficulty 5; priority A

The green/red light pages used in the control rooms at LHO and LLO and also at CIT are popular and something similar should be available for LDR in order for admins to quickly determine if LDR problems exist.

We did not have time to draft a particular design but did discuss some of the information that might be incorporated on such a page. In general such a page should be extensible and it should be easy for new types of information to be added.

This work was given a difficulty of 5 because we believe it will take some amount of work to think clearly about what information to collect and how to present it. Such a web page (or similar) has been decided as absolutely necessary for S4-S5.

Gauge for percentage of collections transferred: difficulty 4; priority B

There should be a simple way for administrators to determine what fraction of a collection (such as 'S5 RDS L3 LHO') has been replicated. This information should probably be displayed on the green/red-light web page mentioned above, but it should probably also be available in other ways either by a command-line or ?

This feature was given a difficulty of 4 since it should not be too difficult to collect the information using the new metadata schema and make it available. We hope to have this available for S4-S5 but it was not deemed absolutely necessary since informed administrators can quickly estimate the amount of data transferred by other means when necessary.

Leverage the BigBrother package for LDR state information: difficulty 7; priority C

The BigBrother package (<http://bb4.com>) may be useful for monitoring the state of LDR, especially for sites such as PSU that already using BigBrother.

This enhancement was given a difficulty of 7 since it would take some investment in time to learn about BigBrother (the UWM group doesn't know much about it) and a priority of C since the green/red-light web page is more important and will go a long ways to providing the information.

Display transfer rates from sites: difficulty 4; priority B

It would be helpful to be able to determine at a glance what transfer rates a site is achieving when replicating data from other LDR sites. Most likely this information would be displayed on the status web page.

This enhancement is not too difficult to add since the information can be easily collected. It was given a priority of B since it is not absolutely necessary for S4-S5 but can probably be implemented sometime during the S5 run.

Easy browsing of data available at site: difficulty 5; priority B

It would be nice to be able to browse what collections of data (S4 LHO level 2 RDS, S3 LLO level 3 RDS, ...) are available at a site. This might be implemented easily using Globus MDS, but it should also probably be available via the web page interface.

This feature request was given a difficulty of 5 since some thought will be needed into exactly what information to display and how to display it and a priority of B since it is not absolutely necessary for S4-S5 but should be present eventually.

Globus scheduling across multiple sites/collections: difficulty 6; priority A

During S3 it happened occasionally that the network to LLO would go down while the network to LHO was still up. Due to the current way that LDR schedules files for replication data would stop flowing from both LLO and LHO while the LLO path was down.

Fixing this problem is straightforward but requires some amount of recoding within LDRSchedule so this task was given a difficulty of 6, but it is absolutely necessary for S4-S5 and was given priority A.

More flexible ordering: difficulty 4; priority A

During S3 and currently the files within a collection such as 'LHO S3 level 3 RDS' are ordered for replication strictly by GPS time. Administrators ought to be able to select from a number of different methods for ordering files within the collection and for ordering how collections are replicated. Ultimately admins should be able to easily order the files using any metadata attribute (GPS time is one such attribute).

This enhancement was given a difficulty of 4 since we have already put some thought into how to implement this request. It was decided that this is necessary for S4-S5.

Archive transfer rate information: difficulty 6; priority B

We would like to archive in the MySQL server transfer rate information for the transfers from each site. There is some concern as to how this would impact the load on the MySQL server and the machine running the server.

We gave this task a difficulty of 6 since it will require a fair amount of work to setup a testbed and explore this idea. It is something we hope to do for S4-S5 but it is not absolutely necessary.

Measure aggregate throughput for file transfers: difficulty 6; priority B

For some sites (in particular MIT) it would be helpful to measure the aggregate transfer rate rather than just the rates to individual sites. This may or may not be best left to some other tool and so we gave it a difficulty of 6 because we will have to explore what tools are available and whether or not they can be/should be integrated into LDR. It was given a priority of B since it is not necessary for S4-S5.

Scheduling based on archived transfer performance: difficulty 6; priority B

Ideally we would like to schedule transfers in a way that takes into account the current network weather so as to be able to move as much data as fast as is possible. To do this we can archive (for some time, perhaps in MySQL or just in memory) transfer performance, but also perhaps use information from other sources.

We gave this task a difficulty of 6 since again it will require exploring a number of different options available to us. It was given a priority of B since it is not necessary for S4-S5.

“On-the-fly” configuration: difficulty 3; priority A

Administrators should be able to make configuration changes to LDR and then have the changes go into effect by sending LDR daemons a HUP (or the like) signal.

This is absolutely necessary for S4-S5 and we gave this a difficulty of 3 since this has already been demonstrated for the current LDRdataFindServer daemon and we only need to incorporate the technology built (a few python classes) into the other daemons.

Rotating log files: difficulty 1; priority A

The LDR log files currently do not rotate and become too long. It is a trivial change to make since we leverage a useful python logging class.

Separate log file for critical error messages: difficulty 1; priority A

A number of admins would like to have CRITICAL error messages logged to an additional log file along with the standard log file for the daemon in order to more easily monitor LDR. It is almost trivial to do this.

Validity checking at server and client end for each transfer: difficulty 6; priority B

It has been requested that LDR verify the integrity of each and every file transfer by comparing a computed checksum to the known checksum. There is concern that this will dramatically slow down transfers, though using some new technology in the GridFTP server and client we may be able to do this by being clever.

This feature request was given a difficulty of 6 since it will require some work to find a way to do this that does not dramatically impact transfer rates. It was given a priority of B since it is hoped that leveraging the LDRVerify work done at UWM by Kevin and incorporating that into LDR will help with the validation testing.

Proxy LDR: If LSCdataFind request fails up the priority for the failed file: difficulty 4; priority C

It has been requested that whenever a request to LDRdataFindServer fails to find a file locally the priority for replication of the missing file be automatically increased. There is concern that this feature could be easily abused and so although not too difficult to implement it was given a priority of C.

Tool for fixing “holes” in metadata: difficulty 3; priority B

Brian pointed out that it is unlikely but still possible that when using the new metadata schema and propagation method a site could develop “holes” in its metadata. It was agreed that it would be very helpful to have a tool to fix these holes should they arise.

Writing up such a tool should not be too difficult so this request was given a 3. It was given a priority of B since the work around as Brian explained should a hole develop is not very painful itself.

Tool for finding “holes” in metadata: difficulty 7; priority B

See the previous note. Brian pointed out that while fixing a hole is not difficult, automatically detecting a hole might be. We need more work to understand this and it might quickly become a priority A.

Slave/master MySQL? difficulty 8; priority B

PSU has investigated using two MySQL servers at a site with one being a slave to the second which is the master. The slave would be optimized for reads to increase performance of tools like framequery and LSCdataFind.

This task was given a difficulty of 8 since it will require a fair amount of testing even though PSU has already done a lot of the work. It was given a priority of B since it is not necessary for S4-S5 data replication.

Data discovery for publishing into LDR: difficulty 7; priority A

Stuart Anderson, Scott Koranda, and Patrick Brady suggested that the diskcacheAPI from LDAS could be leveraged and used in cooperation with LDR (or become a part of LDR?) for data discovery and automatic publishing.

The idea is that the diskcacheAPI, which is very good at watching over a set of NFS or local directories and building a hash table in memory of frame-file locations and limited metadata (the metadata available in the name of the file), could be used to “watch” filesystems and somehow communicate with LDR. This would aid in the automated publishing of data into LDR and also with unpublishing and republishing when data is moved and it is necessary to update the URLs stored within LDR.

Hari in his presentation also discussed how queries to LDRdataFindServer or even framequery could go directly to the diskcacheAPI with a little effort to bend the diskcacheAPI to fit the LDRdataFindServer protocol.

It is agreed that keeping the core of diskcacheAPI, written in C++, the same for its use in LDAS and LDR is desirable. That is, a separate C++ core should not be developed for this new use.

It was argued that the Tcl wrapping around the C++ core that is currently used to interface with LDAS be discarded for this purpose and a new wrapping written, perhaps in Python for use with LDR. The question is definitely still open.

This task is definitely a must for S4-S5 and so was given priority A. The level of difficulty is listed as 7 since a fair amount of work will be necessary no matter how we proceed.

LDRVerify should be easily extensible: difficulty 5; priority A

Kevin Flasch has rewritten LDRVerify at UWM so that it can be used to continually check the veracity of the data. Currently it checks to make sure that data exists, has the correct filesize and the correct uid, gid, and permissions. Soon it will check computed md5 checksums against those computed at the time of publication.

Should this become part of the diskcacheAPI-thing?

Unpublishing files from LDR: difficulty 3; priority A

Hari and others have all written many times “once-off” scripts for unpublishing files from LDR. It should not be difficult to collect these and formalize them into a useful tool.