Experience of using the Chirp distributed file system in ATLAS

Rodney Walker\textsuperscript{1}, Paul Nilsson\textsuperscript{2}

for the ATLAS collaboration

\textsuperscript{1}Ludwig-Maximilians-Universität, München, Germany
\textsuperscript{2}University of Texas at Arlington, TX, USA

Abstract: Chirp is a user-level file system specifically designed for the wide area network, and developed by the University of Notre Dame CCL group. We describe the design features making it particularly suited to the Grid environment, and to ATLAS use cases. The deployment and usage within ATLAS distributed computing are discussed, together with scaling tests and evaluation for the various use cases.

Introduction

The Chirp user-level file system consists of a file server accessible from distributed clients, and is described in detail elsewhere\cite{1}. It is distinguished from traditional file systems, by not requiring root privilege, either on the server or client sides. Also the authentication methods and ACLs are appropriate for a Grid distributed environment. This document describes its usage in a small-scale evaluation project, within the ATLAS Distributed Computing environment.

The primary use-case was to make available the outputs from user analysis jobs on the Grid. It was observed that these files are typically small, a few MBs, and thus the overhead in writing them directly to some remote storage location is negligible. The advantage for the user is that they are immediately available, with no further reliance on Grid infrastructure: Fuse mounting the Chirp area gives very convenient access to the files via direct posix access, e.g. 'cp'. It was also expected that physics groups could make use of Chirp's fine-grained ACLs, to share their data more effectively. ATLAS data in Grid storage, including user data, is readable by the whole collaboration. In principle, voms groups could be used to limit access to say a physics group, but this is not used or implemented – it would be cumbersome. Chirp allows dynamic user specified groups of users(a list of DNs) to have user defined access to file in specific directories.

In order for users to retrieve outputs from a remote site storage, the transfer must use the SRM(Storage Resource Manager) interface, and gridftp. There is a significant overhead, of typically 1 min per file, and this is irrespective of the file size. The client tool uses a few parallel copies to counter this somewhat but, when retrieving 100s of files, there is naturally a lengthy delay. Some users increase this number of parallel transfers, but this can overload the SRM, and degrade other activities.

User don't like this delay but expect it. A more frequent cause for complaint is that the source storage is not working - either broken or in downtime. This situation can persist for days, in the case of a scheduled downtime. Avoiding the SRM altogether, by using Chirp for these small user outputs, avoids both the delay in getting many files and the potential to not be able to get them at all.
Deployment and Integration

Panda Workload Management System

All ATLAS user analysis, as well as official Monte Carlo production, is handled by the Panda system[2]. The relevant features, for Chirp integration, are the pull-model pilot method, and the stage-out to local storage. Empty pilot jobs[3] are submitted via Grid protocols, using one of a few x509 service certificates for authentication. The user payload is pulled from the Panda Db, and runs. The output is stored to the local Grid storage, typically SRM (e.g. dCache or DPM), again using the service certificate. In the current mode of operation, the users personal certificate is not available from the job.

In order to enable Chirp usage, an additional argument was added to the user job submission CLI, --useChirpServer, with argument the hostname of the Chirp server. This triggers a 2\textsuperscript{nd} copy of the user output and log to be stored directly to the Chirp server. The SRM copy is still written, and the job does not fail if the Chirp store fails. This implementation leads to a very low barrier for usage, with no danger of reduced job efficiency.

Authentication

Although Chirp supports several different types of authentication, we have focused on that widely used in other areas of ATLAS distributed computing, namely X509 certificates. Indeed, most Grid environments require users to have an X509 certificate: for ATLAS users it is issued by one of the WLCG trusted certificate authorities. Currently, user jobs, running in the Panda system, run inside multi-user pilots: the pilots run with a common service proxy, and thus the user proxy is not available at this stage. Therefore, in order to write to the user Chirp area, the service proxy needs write access to all user areas. This service proxy is not readily available to the user application, but a malicious user could obtain it and thereby have full access to other users files. He would also have full access to users files in any storage, so this is at least no worse. The multi-user pilot security issue is well-known and can only be solved by identity switching.

Chirp Client and Server Deployment

A single Chirp Server was deployed on a physical node hosted by CERN IT. The specifications were: 8 core (Xeon 5520) with 24GB RAM and a 2TB SCSI disk (not RAID). It was envisaged that institute or regional servers would be setup to provide horizontal scaling: some were, but this was not actively pursued during the evaluation phase. There is at present no mechanism to combine the servers, for example, with a single entry point and namespace. Instead, user jobs would be configured to use one or the other. This lacks simplicity, redundancy and load-balancing, and is a major weakness.

The software installation, via binary tarball, from http://www.cse.nd.edu/~ccl/software/chirp/, was extremely easy: unpack, set the environment variables X509_USER_PROXY and X509_CERT_DIR, then run like

chirp_server -r /data
The configuration is mostly confined to the authentication and ACLs. Due to the Panda multi-user pilot model, the Chirp area must be writable by a small number of service certificates. The user will read and manage their Chirp area using their personal certificate. A user directory has read and write access for the service certificates plus the user's personal certificate. The top-level directory is readable by all ATLAS users, and writable by the service certificates.

The Chirp client provides commands to modify the authentication rules, which are then stored inside a hidden file in each directory, “._acl”. This is not visible when accessing the Chirp area via the client tools, but it can be modified on the Chirp server machine itself. Thus, there is fine-grained, user configurable, access control at the directory level. The ATLAS user database was used to create the user directories and corresponding “._acl” files. This process was not repeated automatically.

The Chirp client is made available to the user jobs, on Grid worker nodes, via the usual ATLAS software distribution mechanism[4].

**Usage and Experience**

Prior to integration into the Panda system, standalone Grid jobs were submitted to test functionality and scaling. Initially, it appeared that there was a severe, and low, scaling limitation, when increasing the number of clients writing files. The number of open sessions increased, and eventually lead to failures. The Chirp host was not under CPU load during this, which excluded the main suspect, namely X509 authentication. Eventually, and with strong support from the developers, it was found that SSL created, and made extensive use of, a file $HOME/.rnd. The HOME area at CERN is on AFS and this turned out to be the bottleneck. Forcing this file to be on local disk solved the problem, i.e.

```bash
export RANDFILE=/tmp/chirp.server.rnd
```

Since integration into Panda, there has been a constant load from 10k file stores per day, written from all over the Grid by some automatic analysis functional tests. Also there is a small, O(10), but enthusiastic group of users who use Chirp as a regular part of their workflow. The feedback from these users is very positive. Being able to access the files directly from Chirp was found particularly useful, for example to merge output files.

Users were very quick to complain about missing files, and several audits were made centrally to check for missing files. None of the cases of missing files could be attributed to Chirp failure, e.g. under load, and we conclude that the scaling limit was not reached during our tests.

The main operational issue - and this did lead to missing files - was keeping the authentication configuration up to date. There were occasionally new pilot submitter DNs, i.e. new service certificates which needed write access. Also there was no automatic synchronization with the ATLAS user database, so new users needed manual intervention. Neither of these is in any way a Chirp problem, but rather due to the limited integration done for this evaluation project. However, Chirp support for VOMS groups and roles would have helped in both cases. For example, we could then give write access to /atlas/Role=pilot.

As the 2TB partition was filled, a periodic cleanup of older files was automated. In this sense...
the demonstration server was operated like a scratch disk, but user/institute servers could set other policies. The quota functionality of the Chirp server was not exercised. The documentation claims there is a performance overhead, in using the quota, as well as a delay in starting the server (while the usage is checked). It was therefore decided to operate Chirp with the same policy as the site storage areas, i.e. with automatic cleaning when full. Figure 1 shows the Chirp disk space usage over the past year: one can see that cleanup was implemented at the end of June.

**Conclusions**

The binary tarball distribution, documentation and examples lead to a very easy deployment of the Chirp server. This is well-suited to institutes and even individual users setting up the Chirp service on a desktop, say. Similarly the client tools, particularly the Fuse mount, were very easy to use and convenient for the users.

The service was very stable, and essentially maintenance free over the 1 year period of running. Failures were rather due to lack of production-level integration, inherent in such an evaluation project. These included authentication configuration updates, and service restart after unscheduled reboot. It was noted that VOMS integration would ease the configuration and maintenance, and is probably needed for wider HEP community use.

Unfortunately, the test scale did not reach the limitation of a single Chirp server. There are clearly trivial limits from bandwidth and disk performance, but it would be good to know the limits from Chirp itself, and this is a subject of future work. Horizontal scaling with further Chirp servers is envisaged. This could take the form of institute or national instances to provide the Chirp service for a group of users. Although Chirp is usable today and performed well, it is expected that scalability requirements and the attractiveness of using standard protocols will lead to a better solution, perhaps using webdav.

**References**

1. [Chirp: A Practical Global Filesystem for Cluster and Grid Computing](https://doi.org/10.1007/s10723-008-9100-5), Douglas Thain, Christopher Moretti, and Jeffrey Hemmes, Journal of Grid Computing, 7(1), pages 51-72, March, 2009. DOI: 10.1007/s10723-008-9100-5
2. Overview of ATLAS PanDA Workload Management (slides, paper), T. Maeno et al., 2011 J.Phys.:Conf.Ser. 331 072024
3. Recent Improvements in the ATLAS PanDA Pilot, P. Nilsson, Proc. of the 19th Int. Conf. on Computing in High Energy and Nuclear Physics (CHEP 2012)

4. The ATLAS software installation system for LCG/EGEE, A De Salvo et al, 2008, J.Phys.:Conf.Ser.119 052013