Xrootd Storage Element Deployment at GridKa WLCG Tier-1 Site

Artem Trunov
Karlsruhe Institute of Technology,
Kaiserstrasse 12, 76128 Karlsruhe, Germany
E-mail: artem.trunov@kit.edu

Abstract. In the work we present our effort in designing and building of Petabyte-level storage element based on Xrootd with storage backend and SRM integration. This Storage Element is implemented at the German LCG Tier-1 Center “GridKa” for the ALICE experiment. Motivation, use cases and details of implementations are presented. It’s shown that Xrootd SE at GridKa has no single point of failure and satisfies the initial requirements.

1. Introduction
Xrootd is a data access system developed at SLAC National Laboratory in collaboration with CERN and other institutions worldwide [1]. Xrootd is a part of ROOT package which is used extensively in High Energy Physics community and other fields of science [2]. It is also distributed independently from the ROOT framework. ALICE experiment of the Large Hadron Collider relies on xrootd for all data storage and access and requires sites supporting ALICE computing to deploy xrootd-based storage elements [3]. To make site administrators’ job easier, CERN provides xrootd in a special bundle that can be installed, configured and operated with minimal efforts. However this distribution mostly targets smaller T2 sites and doesn’t include support for large T1 site requirements. Typical T1 role implies: import of raw and processed data from CERN for custodial archiving on tapes, serving raw data for subsequent reprocessing, storing locally produced reprocessed data on tapes, serving data for analysis. Thus tape back-end and SRM interface are required in addition to the basic xrootd package. SRM is a Storage Resource Manager – a protocol for storage and transfer management, adopted by WLCG [4]. SRM uses GSI-based authentication.

The aim of this project was to design fault-tolerant xrootd-based storage element for ALICE with tape backend and SRM frontend. Reliability and low maintenance was a special focus of this project.

Gridka has been dealing with xrootd since 2002 when it was installed to support the BaBar experiment. In November 2008 a disk-only xrootd system with 320 Terabytes was installed for the ALICE experiment. In September 2009 a 480 Terabytes Xrootd SE with tape-backed and SRM front-end was set up for the ALICE experiment. Since November 2009 ALICE is using Xrootd SE exclusively and the size of the total space deployed in disk-only and tape-backed Xrootd SE instances is over 2 Petabytes.

2. Storage setup at GridKa T1 center
To store scientific data, GridKa T1 center employs IBM’s General Parallel File System (GPFS) setup on standard x86 Linux servers with Fibre Channel storage attached to them, either directly or via Storage Area Network (SAN) [5]. In either architecture, there are redundant data paths. In case of the
direct-attached Fibre Channel solution, storage arrays have redundant controllers and each server is connected to both controllers. Thus if one controller fails, the system stays in production because the other controller continues to function. In case of the SAN solution, servers are connected to two SAN switches that are connected to redundant disk array controllers. Thus, additional redundancy is achieved via a SAN fabric.

To achieve even higher availability of the hardware, each disk controller is connected to at least two servers, so that a failure of a server doesn’t result in data unavailability. Server failover mechanism is implemented in GPFS. Both servers participate in so called GPFS cluster, and both mount the same file systems. Thus the same data files can be accessed via any server participating in the a cluster. On Pic. 1 these storage setup options are illustrated.

3. Xrootd storage Element
The storage setup described in the previous section matches very well the operational model of xrootd. Xrootd consists of server daemons that provide access to data, and manager daemons that handle user’s initial requests and organization of server daemons in a cluster in order to enable a single name space for files, load balancing etc. In our setup, xrootd server daemons run on GPFS servers, and provide access to all data files in all file systems mounted on GPFS servers. A single file can be access via all servers in the GPFS cluster. This helps to achieve high availability of data, and also spreads the load between all servers in the cluster in case of access to a “hot” file – a file that is requested by a large number of clients at the same time. This case is realized for conditions data files that need to be opened by many jobs processing different parts of one data set. Redundancy and failover mechanism is illustrated in Pic. 2.

4. Tape backend for the Xrootd Storage Element
Xrootd itself has support for tape backend from the very beginning of its life. Tape back-end related components include: policy-based migration daemon, policy-based purging daemon (aka garbage-collector), prestaging daemon, migration and purging queues (supporting 2-level priority), on-demand stage-in while user’s job wait on file open, bulk “bring-online” requests, asynchronous notification of completed requests via UDP messages. Current “mps” scripts providing tape back-end functionality are being replaced with more feature-rich and robust File Residency Manager or “frm”.

Picture 1. FC direct connect and SAN options
Picture 2. Redundancy and failover with xrootd
Picture 3. Tape backend and “cloud” connection
The mechanism of integrating a site-specific Mass Storage System is very flexible. It is only necessary to implement own glue scripts – a “stat” command to get basic file info from the Mass Storage System, and a “transfer” command that calls MSS’s own tool to copy file in/out of the MSS.

GridKa uses IBM’s Tivoli Storage Manager as a tape archiving system and in-house developed middleware to control migration and recall queues – Tape Staging System (TSS) [6]. This is the same mechanism used by dCache SE at Gridka.

TSS setup in Xrootd SE is as following. One of the nodes in a GPFS cluster is connected to tape drives via Fibre Channel SAN. This node stores all files from the GPFS cluster to tape and recalles all files from tapes for the whole cluster, evenly distributing them across all GPFS file systems in the cluster. This helps to reduce the number of tape mounts, since TSS can optimize tape mounts within one server only and employing the second server could result in mounting the same tape twice – first for files requested by one servers, and second – for files requested by the second server.

TSS mechanism works in conjunction with ALICE’s cloud, implemented via a “global redirector” combining all ALICE sites in a single meta-cluster. In this meta-cluster, a file at any site can be accessed via a single entry point – “global redirector”.

When a file is not on disk in GridKa Xrootd SE, it will be first looked up in the ALICE cloud and then in the local tape archive, thus giving priority to site-to-site network transfer, which is considered “easier” then tape recalls, and tape system plays its original permanent tape backup function.

Tape backend and cloud connection are illustrated on Pic 3.

5. Grid Front-end for the Xrootd Storage Element
Grid front end development efforts were spent mostly due to requirements of the SRM interface, adopted as a standard in the WLCG project. While ALICE experiment’s middleware makes no use of it in production, it is still needed to ensure the WLCG requirement.

To implement grid front end in the Xrootd SE at GridKa, existing OSG solution was used as a baseline. However it could not be used out-of-the-box, because of specific differences between OSG and gLite middleware organization, especially in packaging and authorization areas. But the components of grid-enabled Xrootd SE are all the same: Xrootd itself, Globus gridftp server with posix dsi backend and xrootd posix preload library for data transfer, xrootdfs for filesystem view of files stored in Xrootdfs, and BeStMan SRM server in gateway mode implementing the SRM protocol.

Grid components and their interaction are shown on Pic 4.

6. Implementing Xrootd SE at GridKa
This section provides details of development efforts spent implementing Xrootd SE at GridKa.

Xrootd package came from CERN repository created and maintained by Fabrizio Furano. This repository is the official repository used by all ALICE sites installing native xrootd storage element. It provides ease of configuration and operation – topics that are very much appreciated by site’s administrators. Xrootd package was installed and configured according to ALICE wiki page with little or no deviations.

To install Gridftp server, VDT rpms were used. VDT packages gridftp server from patched Globus 4.2.1 version. Gridftp posix preload library was also taken out of the OSG repository and packaged into an rpm for convenience of deployment.

Host certificates, CA certificates and revocation lists, and the gridmap file were all installed following standard procedures developed for a typical gLite environment. With the gridmap file, we had to make sure that the DN used for transfer is mapped to a static account (as opposed to a group pool account) since BeStMan SRM only supported static accounts or the OSG’s GUMS authorization plugin. Xrootdfs was compiled from source obtained from the official repository at SLAC, and rpm was made for convenience of deployment. Fuse libraries – prerequisite for xrootdfs – were already a part of Scientific Linux 5.5 distribution. BeStMan was obtained as a tar file from the official
repository at LBNL [7] and configured using supplied configure scripts and an excellent user manual. Information system entry (BDII publishing) was made using a static description file in the glue schema version 1.3 format.

All components except xrootd daemons were run as root user to avoid complications.

7. Redundancy in the Xrootd SE’s admin nodes
In the second and this sections we have shown how redundancy was achieve on the server level. With the described setup, any server can fail at any time and the failure doesn’t result in reduced uptime or inaccessible data. This means that maintenance can be done without taking the whole SE offline, without announcing downtime, just by performing so called rolling upgrades – one server at a time.

In this section we describe how admin services were made redundant. Admin services are xrootd manager daemons, BestMan daemon, Gridftp and xrootdfs daemons, as illustrated on Pic 5.

Admin services were all setup in virtual machines running on one of two servers with KVM hypervisor. Virtual machine (VM) images were stored in a GPFS file system mounted on both servers. Thus, in case of a failure of one VM host, guest VMs could be easily restarted on the other server. GPFS file system was configured to create two replicas of each block for enhanced redundancy.

Xrootd manager daemons were configured to run in a dual-manager mode. In this mode, two in two separate VMs with manager daemons were run under a single DNS name alias. Xrootd protocol supports this mode natively by resolving the DNS name alias and selecting a manager node that is up at the moment of the request. Each guest VM with an instance of xrootd manager is hosted on a different VM host, so that in case of failure of one of the two VM hosts, one instance of xrootd manager is always online. In case of system maintenance on VM hosts, it can be done on one server at a time, live-migrating or restarting its guest VMs on the other server.

Gridftp daemons were setup in the so called split-process mode, in which there is a front-end daemon that establishes a control channel for commands and a back-end daemon that is actually moving data buffers (see Pic. 5). There are several advantages in such a setup. Only gridftp front-end daemon needs a host certificate, while several corresponding back-end servers don’t. Back-end server daemons can be run on actual physical data servers, while front-end can run in a virtual machine. Last, back-end and front-end servers can be optimized for their tasks: front-end server for low-latency network communication, while back-end for large block data movement.

8. Performance of the Xrootd SE
While we have no dedicated test bed for systematic performance measurement, we can report some observation of the performance in real production. The following pictures illustrate transfer performance and tape archiving performance. Picture 6 shows production transfer of raw data from ALICE T0 buffer at CERN to GridKa Xrootd SE. Using a cluster of three servers, sustained rate of 450 Megabytes per second is achieved, peaking at over 500MB/s. Picture 7 shows sustained archiving to tape from one server in the cluster. Rate of 300 MB/s was achieved using up to 4 tape drives in parallel. The numbers are well within nominal throughput required by ALICE computing Model.
9. Conclusions
We presented an Xrootd Storage Element setup at Gridka T1 center. We have shown how different components are set up to achieve a fully redundant system with no single point of failure. We have reported some performance numbers observed in real production that fall within design goals.

References
[1] Alvise Dorigo, Peter Elmer, Fabrizio Furano, Andrew Hanushevsky, "XROOTD/TXNetFile: a highly scalable architecture for data access in the ROOT environment", Proceedings of the 4th WSEAS International Conference on Telecommunications and Informatics, 2005, ISBN:960-8457-11-4.
[2] Furano F. Data Management in HEP: an approach. To appear in The European Physical Journal - Plus (EPJP), 2011
[3] Betev L, Carminati F, Furano F, Grigoras C, Saiz P. The ALICE computing model: an overview. Third International Conference "Distributed Computing and Grid-technologies in Science and Education", GRID2008, http://grid2008.jinr.ru/, 2008
[4] Lana Abadie, Paolo Badino, Jean-Philippe Baud, Arie Shoshani, Alex Sim, et al.: Storage Resource Manager version 2.2: design, implementation, and testing experience. In: Conference for Computing in High Energy and Nuclear Physics, 2007
[5] Data storage layout and architecture of the German T1. Ressmann, D., Böhringer, S., Halstenberg, S., Van Wezel, J.Internat.Conf.on Computing in High Energy and Nuclear Physics (CHEP 07), Victoria, CDN, September 2-7, 2007
[6] TSS, an interface to TSM for storage-management middleware. Jos van Wezel. Oxford University TSM Symposium, 25-27 September 2007
[7] Berkeley Storage Manager: http://sdm.lbl.gov/bestman/