CMST SI(1) 109-118 (2010) DOI:10.12921/cmst.2010.SI.01.109-118

COMPUTATIONAL METHODS IN SCIENCE AND TECHNOLOGY **Special Issue 2010,** 109-118

Popular Backup/Archival Service and its Application for the Archival of the Network Traffic in the PIONIER Academic Network

Maciej Brzeźniak, Norbert Meyer, Rafał Mikołajczak, Gracjan Jankowski, Michał Jankowski

Poznań Supercomputing and Networking Center
Institute of Bioorganic Chemistry, Polish Academy of Sciences
Noskowskiego 12/14, 61-704 Poznań, Poland
e-mail: {maciekb/meyer/rafal.mikolajczak/gracjan/jankowsk}@man.poznan.pl

(Received: 27 September 2010; revised: 8 November 2010; published online: 23 November 2010)

Abstract: This paper presents the popular backup/archival service developed and operated in Poland by members of the PIONIER network consortium and its example application for outsourcing of the archival of the network traffic in the national academic network. The service is built upon the National Data Storage (NDS) system architecture deployed in the redundant, high-end, geographically distributed infrastructure of servers, network and data storage systems built within the confines of the PLATON project. The details of the NDS architecture and its features are discussed in the paper including the system components, their functionality and the system scalability aspects. The paper also presents how the NDS architecture is deployed in the data storage infrastructure of the PLATON project, with an extensive usage of servers and storage virtualization technologies. We discuss how the NDS system instantiation allows for flexible set up of the multiple instances of the popular backup/archival service, which can address various, often contradictory requirements of the service users, while sharing a common pool of physical resources. As an example the system set up for outsourcing the archival of the PIONIER network traffic is presented.

Key words: data management, data archival, backup, virtualization, distributed data storage

I. INTRODUCTION

Storing massive amounts of data for backup or archival purposes is a challenging task both for institutions and individual persons. It is projected that 1800 exabytes of data it is going to be produced in 2010 across the World. The estimated amount of data produced per person in 2010 may reach 260 gigabytes, according to IDC [IDC1]. Moreover, amounts of data produced and processed in today's computer systems exceed the storage capacity available in storage systems, and the gap is still enhancing.

Managing, classifying and storing as well as short-term and long-term protecting these data are complex and expensive processes, as they require know-how and resources including man power and money needed for the purchase and maintenance of the infrastructure.

A lot of massive data producers represent the scientific and academic environments. The main 'data generators' in Poland include digital libraries, virtual laboratories, academic computing centers and academic network operators as well as scientific institutions such as research institutes (600+ organizations), universities (almost 500) and clinical hospitals (more than 50). Even conservative simulations suggest that these institutions and projects may need storage space for backup and archival purposes expressed in hundreds of terabytes per year.

An example of an institution that generates such big data volumes is the operator of PIONIER, the Polish national academic network [PIO1]. Legal regulations that apply to network operators force them to store some percentage of protocol headers of the traffic traversing their network, and be ready to present these data to authorities for investigation purposes. The PIONIER network carries about 20880 TB of the traffic per month. Assuming that the network operator is forced to record protocol headers of 5% of the total traffic, about 14 TB of the storage space per month is required for that purpose. This ends up in a need for storage space about 168 TB per year and half of petabyte after 3 years.

Scientists and students themselves can add a considerable volume to this "national data storm", taking into account that more than 100 thousands people are employed in scientific institutions in Poland [GUS1] and almost 2 million students attend universities [MIN1].

Having said that, we clearly see that protecting and archiving the data may exceed the abilities of individual institutions, projects and persons. Moreover, dealing with data protection is not the core business of the academic or research institutions and individual scientists, so the wish and ability to designate considerable resources for that purpose is not common. Therefore, outsourcing the process may be the most reasonable solution.

Popular backup/archival service is a response to the above-mentioned needs. It is intended to free the end-users' mind from dealing with short-term data protection and long-term data archival. The service allows the academic and scientific institutions and individuals to store, keep and maintain the data according to some agreed policies and service profiles.

In this paper we discuss the architecture, deployment and example use case of the popular backup/archival service. The architecture of the service is based on the outcome of the R&D project "National Data Storage" [NDS1]. The basic architectural decisions taken during the NDS system design and development are discussed in Section II of this paper. We focus on these aspects of the architecture that address critical user requirements gathered in the early phase of the project. In Section III we show how the NDS system architecture and the software stack is deployed in the data storage infrastructure composed of redundant and geographically dispersed servers, tape libraries, disk arrays, file servers and storage management software supporting the virtualization technologies. This section also shows how the NDS architecture and the redundant infrastructure of the PLATON project address the user requirements by making it possible to provide the reliable, secure and scalable data backup and archival service. In Section IV we present an example use case of the data archival service, i.e. recording the network traffic headers in the PIONIER network. Finally, we discuss the related work in the area of distributed storage systems for data backup and archival, and we conclude our study in the "Summary" Section.

II. SYSTEM ARCHITECTURE, FEATURES AND FUNCTIONALITY

The main assumption of the National Data Storage project was to design and develop the data storage service targeted to data backup and archival storage applications. Various approaches were possible to implement the distributed data store, depending on the most important features expected from the system. Architectural decisions that we have taken were motivated by the following factors.

Firstly, we assumed that high-availability and reliability, including data durability, require a geographically distributed storage system with data replication. Distribution of the architecture is also necessary to provide system scalability in terms of performance, storage capacity and the number of users. We were, however, aware of the challenges the distribution brings while trying to guarantee consistency, fault tolerance and high performance in the system.

Secondly, we decided to focus on specific system features and functionality. In order to target actual user needs, we collected detailed user requirements against the features and user interface of the service in the early stage of the project (i.e. the autumn of 2007). The survey results confirmed that data durability and service availability are the most important features demanded by users. Half of the users explicitly expressed the need for geographical data replication. Replication was also requested implicitly as the consequence of availability requirements, most of the potential clients indicated. Features-wise, no users expected data sharing or exchange capabilities from the system. On the contrary, most examined institutions wanted to have dedicated name spaces in the system and required the confidentiality of the data.

Thirdly, we had to be realistic about the features and functions we are actually able to provide as a reliable and stable production-level service, having in mind the available budget and the time schedule. We expect users to charge the system with a task of protecting and archiving their data, and therefore the reliability of the system was of the highest importance.

II.1. System architecture

Having in mind the above factors, we have worked out the architecture of the National Data Storage system (Fig. 1), that combines distribution and centralization of selected components in order to meet contradictory requirements and expectations we had to deal with.

The data are stored in a fully distributed manner in multiple, geographically dispersed Storage Nodes of the system, which assures the data durability. Service reliability and high-availability is guaranteed by the fact that at least one Storage Node is always able to serve the data replica or accept the incoming user data, nevertheless the failures of another Storage Nodes in the system.

A user can access the data through multiple, distributed Access Nodes. This again improves the service's availability as the fail-over between Access Nodes is possible in case when part of the Access Nodes is unavailable due to emergency or scheduled downtime. Moreover, the existence of multiple Access Nodes assures the system scalability in terms of data access performance as well as the number of users and institutions served effectively.

Meta-data DB that maintains the file system structure as well as replica information and another data objects' meta-data is logically centralized in order to keep the consistency, fail-over and disaster recovery techniques simple and reliable and make the meta-data operations effective (e.g. no need for distributed locking). Physically, the Meta-data DB is redundant. Meta-data are replicated from master Meta-data DB to a set of slaves. This eliminates a single point of failure and allows the roll-back and recovery in case of emergency. The Meta-data DB redundancy also assures the meta-data durability, which is critical from the point of view of the system reliability.

Potentially, single Meta-data DB can impose the limitations on the meta-data handling efficiency throughput and the number of data objects stored in the system as well as on the overall amount of the meta-data kept for the data objects. However, multiple meta-catalogs can exist in the system and serve particular, distinct institutions or groups of users. Such an approach is acceptable, as data sharing among organizations is not expected by potential system users. The solution is described in details in the 'Architecture Deployment' section of the paper, where we discuss how the instantiation of meta-catalog as well as other system databases and components can be used in order to omit possible limits of system centralization.

II.2. Basic system components

The NDS Virtual File System is the interface between the NDS system logic and the Access Methods exploited by users in order to store and retrieve data and meta-data (see Fig. 1). The NDS system logic itself is implemented as Data and Meta-data Daemons (not visible in the Fig. 1) using the FUSE library that allows to develop the file system drivers at the user level [FUSE1]. NDS system daemons serve the virtual file system requests such as create, read, write, flush, release etc., by performing actual data operations on the distributed data replicas and making meta-data operations in the meta-data DB. Data replicas are stored and retrieved by the Data Daemons in the Storage Nodes using Replica Access Methods, including NFS and GridFTP (see Fig. 1) protocols, while meta-data are stored and maintained in the meta-data DB.

Meta-data DB holds the critical information about the virtual file system structure. It includes names of logical files and directories, logical data objects hierarchy (directories, subdirectories and files), mappings of logical files to physical replicas as well as logical file and directory attributes, including standard file system attributes such as file size, modification and access times as well as access and ownership flags. Additionally, meta-data DB keeps the replica information including physical replica location, attributes as well as additional NDS-specific attributes of logical files and directories such as annotations and data objects history (operations record).

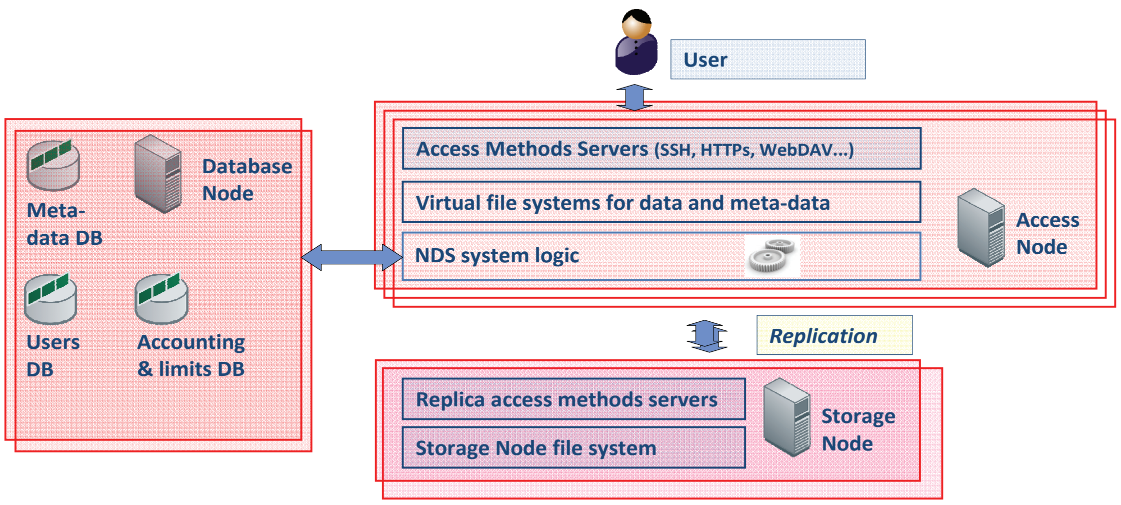


Fig. 1. Overall architecture of the system

The meta-data DB is implemented as a PostgreSQL data-base [PSQ1]. Fault tolerance of the meta-catalog is assured by a semi-synchronous meta-data replication. The database transactions are asynchronously replicated using Slony-I for PostgreSQL [SLO1]. Additionally, each meta-data operation is synchronously recorded in multiple, distributed operation logs. In case of emergency, a slave copy of the meta-data DB can be used as the master DB, as it contains an asynchronously created copy of the master DB. In case when this asynchronous copy does not hold the consistent state, consistency can be recovered by using distributed operation logs reflecting the synchronously updated meta-data operations history.

Beside the meta-data DB, the system and system services operation is supported by Users, Accounting and Limits Databases (see Fig. 1). Users DB contains the profile information for the users of the service, including the data replication mode (synchronous versus asynchronous), a desired number of replicas and allowed replica locations as well as other service parameters such as resource usage limits etc. Accounting DB stores history of the resources usage (mainly the used storage space) at the level defined for particular users. Fault tolerance of these databases is assured, by supporting fail-over from the failed master database to the asynchronously updated slave. As the data are mainly read from the Users Database, asynchronous database replication is acceptable. Similarly, replicating Accounting and Limits Databases with this mechanism does not bring significant risks, as these data are not critical for the system operation nor users' data durability or consistency.

The National Data Storage system operation is supported by auxiliary mechanisms, not visualized in Fig. 1. For high-availability reasons, the responsiveness of the system components including servers, daemons and databases is constantly monitored and taken into account by the NDS Data Daemon while processing the users' I/O requests. Failures are also reported to the system administrators using a monitoring portal, e-mails and other communication channels. Additionally, the load of disk arrays, tape systems (status of queues, tape robots and drives) and other system components are monitored for performance optimization purposes.

II.3. System functionality and interface

Data replication. The NDS system performs the automatic data replication, that is transparent to the end-users. When asynchronous replication is used, writing the data to the virtual file system is confirmed immediately after the successful writing of at least one replica to any of the Storage Nodes allowed for a given user. This allows to minimize the time needed to store the data into the system, observed from the user point of view, while assuring that the user's data is protected against the single replica failure after some period of time, needed to asynchronously create additional physical data object instances. Synchronous data replication is also supported. In this mode, storing the data requires more time, from the user's point of view, however, it is guaranteed that all modifying I/O operations requested by the user are immediately performed on all replicas of the data and finished when the completion confirmation is sent to the user.

The system interface is composed of two main elements. The Virtual File System interface (see Fig. 1) provides the low-level, file system-like data storage and retrieval functions that abstract the data management operations performed inside the system such as data replication, meta-data processing, operation logging, accounting and limits implementation and system components' failures handling. Secondly, these low-level functions are made available to users through the Access Methods (see Fig. 1). This assures another level of data storage and access abstraction – users have a feeling of interacting with remote SFTP, HTTP, WebDAV or GridFTP sites, and do not have to care about the services implementation details or the physical data location. From the service provider's point of view, the abstract logical file system interface helps to maintain the system security and availability. Performing updates or security patches to the Access Methods software is not problematic as these methods interact with the storage system through a standard file system interface.

III. ARCHITECTURE DEPLOYMENT

In the previous section we have presented the architecture of the National Data Storage system that combines the distribution and centralization of selected components. We have also indicated possible scalability issues related to the logically single meta-data database. In this section we discuss in detail the scalability of the single system instance and show how bottle-necks of the partially centralized system can be omitted by using multiple instances of the whole system or its selected components deployed in the virtualized infrastructure of servers and storage resources.

III.1. Single instance scalability

We have shown above, that the scalability of the single system instance is supported by logical and physical distribution of the multiple Storage Nodes and Access Nodes. We have also noted that the system performance and storage capacity can grow up to the point in which the logically central meta-data DB becomes the bottle-neck. Below we comment on the system scalability in detail, in the context of the efficiency of the “throughput-intensive” versus “meta-data-intensive” I/O operations as well as the system capacity versus the number of data objects that can be stored in the system.

Numerous Storage Nodes enable to scale data access performance and the system capacity, in addition to obvious availability and reliability advantages. Data transmission throughput of the system increases with the growing number of storage sites as the data traffic can be distributed among them. Similarly, the responsiveness of the system is better if more Storage Nodes can serve the data access requests in parallel. Note, that monitoring and requests execution optimization techniques are in place in the National Data Storage system. Capacity of the system, both in terms of storage space and the number of files stored in the system, increases with a growing number of the Storage Nodes and storage devices (disk arrays, tape system or HSM systems) connected to these nodes.

Similarly, multiple Access Nodes help to scale the system performance and capacity, in addition to availability and reliability advantages. Distribution of the user load among a high number of Access Nodes improves the data throughput and responsiveness to the I/O request in the system. Numerous streams of data incoming to the system or retrieved from it can be served in parallel by distinct instances of Virtual File Systems running on multiple servers. Additionally, parallelization of the users’ data access sessions allows to compensate communication latencies caused by the need of exchanging of the I/O requests and user data between the user-level daemons and the VFS layer of the Linux kernel. Note, that the NDS system logic is implemented on the user level (opposite to kernel-level file system drivers) and the efficiency of the communication between user and kernel layers in a single operating system can be limited. In terms of system capacity, the existence of numerous Access Nodes allow to overcome possible capacity limits of single Virtual File System instances, related to Linux VFS layer limits.

While distributed Access Nodes and Storage Nodes enable the scalability of the system, the logically centralized meta-data DB (called also meta-catalog) can be both capacity and performance bottle-neck.

Capacity limits imposed by the meta-catalog can affect the maximum number of the data objects stored in the system. For directory, at least one data base entry describing its name and attributes (both standard and NDS-specific) is required. For file, the replica information also has to be maintained, which requires at least one database record in the meta-catalog for each file replica. In terms of database capacity itself, the number of database entries should not be an issue, as PostgreSQL supports the tables of size up to 32 TB and does not impose limits on the number of rows per table. However, we can expect the performance degradation happening after exceeding some database size threshold. Note, that the meta-catalog does not impose any limits to the storage space available in the storage system, as its possible impact relates to the number of data objects in the system, not to the storage space they can use.

Processing meta-data transactions in the meta-catalog can be a source of performance limits of the single NDS system instance. Application-level operations such as storing the file in the system are in practice composed of numerous file-system level actions such as directory lookup (*lookup()*) and attributes checking (*getattr()*), creating the file (*mknod()*), opening the created file (*open()*) and, finally, performing a sequence of writing (*write()*) and flushing (*flush()*) operations, followed by closing the file and releasing the descriptors (*release()*). File system-level I/O operations, in turn, can require at least some queries to the meta-data database tables or even their modifications. For instance, opening the file in a read-only mode, requires assigning the database-level handlers and applying the locks. It also generates history record(s) and can result in other modifying operations to be performed in the database. Moreover, most actions performed on the logical files requires corresponding actions on the physical replicas which can produce another portion of database transactions.

Overall, the user actions that require creation, deletion, renaming or changing the attributes of the file impose a significant load of the meta-data database. Therefore, the responsiveness of the NDS system to these meta-data intensive operations can be impacted by a limited throughput of the logically centralized database. On the other hand, reading the content of the files or writing data to them is not vulnerable to meta-data handling issues that can happen due to meta-catalog centralization, as these operations are not meta-data intensive.

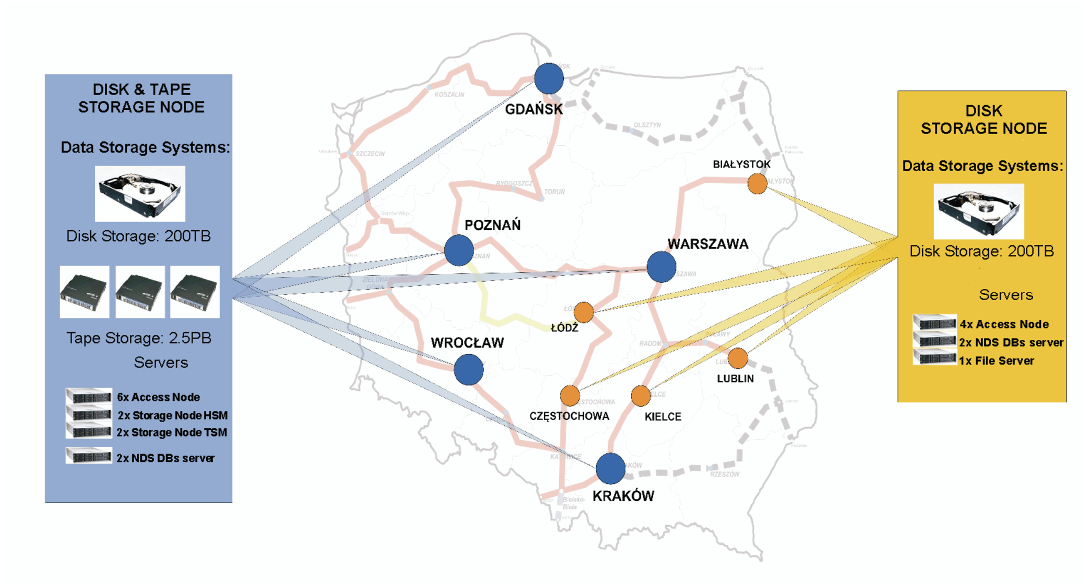


Fig. 2. System deployment infrastructure

Note, that in the backup/archive applications, the majority of the I/O traffic incoming to the storage system contains the sequential write or read operations with a large requested data block size. Therefore, we believe that the proposed architecture of the NDS system is suitable for the backup/archive service, as it assures the performance scalability of the throughput-intensive data operations.

III.2. System instantiation

Potential limitations of the NDS architecture related to meta-data processing can be omitted by running multiple instances of the system in a redundant, distributed infrastructure of servers, storage devices and database management systems that support the servers and storage virtualization technologies.

Such infrastructure, along with the NDS architecture features, allows flexible deployment of the system components in order to meet both the end-user requirements and the infrastructure owner's wish to keep the system configuration simple and the need to use the resources effectively.

As the meta-data handling is the potential limitation of the NDS architecture, the deployment infrastructure, should enable to configure separate meta-data databases for particular system instances. Such approach is in line with the assumptions related to the system functionality, as no data sharing between distinct user groups (e.g., distinct institutions) is required. Particular meta-data DB instances can be created on dedicated physical or virtualized servers or they can co-exist within the same database management system, depending on the expected or actual user requirements against the meta-data processing efficiency.

The deployment infrastructure should also allow to assign pools of Access Nodes and Storage Nodes to particular system instances. These nodes can again be, configured on dedicated physical or virtual servers depending on user needs against the data access performance and volume of the data stored by them in the system.

The deployment infrastructure for the NDS system (see Fig. 2), which is built in the confines of the PLATON project, meets the above-mentioned requirements. It is composed of multiple, geographically dispersed servers, tape libraries, disk arrays and file servers. Virtualization technologies and storage management software such as the VMware ESXi [VMW1] system and the HSM systems [HSM1] are also present. The infrastructure is funded from the EU structural sources under the PLATON project (Service Platform for e-Science, POIG.02.03.00-00-028/08).

Flexible configuration of the PLATON infrastructure elements and seamless deployment of the NDS system instances is possible thanks to the physical and logical infrastructure redundancy and distribution and the fact that the infrastructure supports the servers and storage virtualization. This assures that critical users' requirements such as data access performance as well as service reliability and data security can be addressed.

Particular NDS system instances set up in the PLATON infrastructure can share selected physical resources, or can be run on dedicated resources depending on the end-users needs against the data and meta-data processing performance as well as data and meta-data capacity.

For instance, particular NDS Virtual File System modules and Data and Meta-data Daemons can be run in dedicated Access Nodes belonging to distinct NDS system instances. Such approach is reasonable if the user is going to store a considerable volume of data or a big number of files in the system, and expects the data access to be effective. The servers can be dedicated to NDS instances in a logical or physical sense. In the former case, multiple virtual machines running the Access Nodes are run in a shared pool of physical servers, by exploiting the functionality of the server virtualization platform present in the infrastructure. In the latter case, one may benefit from the fact that multiple physical servers exist in the environment, and thus using a subset of these servers as Access Nodes for a selected NDS instance is possible. Note, that both solutions are supported by the NDS system architecture.

On the other hand, if the expected data traffic generated by the user is moderate, the Access Nodes exploited by a given user can be run in the physical servers shared with another system instances. Sharing of the servers can happen in two ways. Separate virtual machines can be used to run Access Nodes belonging to distinct system instances. Alternatively, the NDS Virtual File System or Data and Meta-data Daemons as well as Access Methods belonging the distinct NDS instances can share selected virtual machines. Again, the NDS architecture and PLATON infrastructure supports both approaches.

The distinct NDS instances, while having dedicated or shared Access Nodes, can use dedicated or common pool of Storage Nodes providing separate NFS exports. Separate exports can by used for storing data replicas belonging to distinct NDS instances. Both setups are possible in the infrastructure and are in line with the NDS system architecture. The decision whether to use dedicated or shared Storage Nodes depends on user needs related to replica access throughput. Storage virtualization technology such Storage Area Network and features of the disk arrays (flexible RAID and LUN configuration) and the file servers (flexible NFS shares setup) belonging to the infrastructure, help to provide appropriate storage resources to particular Storage Nodes comprising the NDS instances.

When meta-data processing performance is considered, two approaches to deploy NDS architecture in the PLATON infrastructure are possible. If the user requirements against the meta-data processing performance is moderate, multiple NDS system instances can share a common database management system (PostgreSQL in our case) hosting multiple, logically separate instances of the meta-data databases. However, if the user is expected to generate a lot of meta-data, distinct NDS instances can use dedicated meta-data database nodes. Again, separation of the NDS system nodes can be performed in the logical sense (multiple database virtual machines running on a shared pool of physical servers) or in the physical sense (a dedicated physical server for a given DBMS).

The decision, whether to use shared physical infrastructure elements for distinct NDS instances (and how to share them, i.e. by using servers or storage virtualization) or to use the physical resources dedicated for particular system instances, should be taken (while) having in mind the user requirements against the data access performance as well as their expectations related to the service reliability and the data security. However, also the user expectations related to the service reliability and the data security should be considered.

If the end-user requires high reliability, multiple physical elements, such as Access Node, can be assigned to host virtual machines implementing the storage service for this end-user. Note, that thanks to server virtualization technology, dedicating these physical servers to the NDS instance is not necessary. Reliability is guaranteed by the existence of multiple logical Access Nodes and Storage Nodes in the system instance, even if physical servers on which the logical servers are run are shared with another system instances.

In case of extremely high security-related requirements, a dedicated system instance can be used for the user. Elements of this instance can be isolated from the other instances on multiple levels, by running them on separate virtual machines, configuring separate virtual Ethernet, IP and SAN networks (by using VLANs, VPNs and SAN zoning technologies) and dedicating cryptographically-protected storage devices such as encrypted tape pools and disk volumes. Note, that the PLATON infrastructure components supports the features necessary to configure a securely isolated NDS.

Overall, the redundant PLATON infrastructure allows for flexible deployment of the NDS system instances, in the way that addresses various and possibly contradictory user needs. The NDS architecture itself supports different configuration options, so the production services built upon it can benefit from advanced deployment infrastructure features such as servers and storage virtualization.

IV. SYSTEM USE CASE

In the previous sections we have presented the NDS system architecture and shown how it can be deployed in the data storage infrastructure built in the confines of the

PLATON project, in order to address the end-users' needs on performance, reliability and security of data storage and retrieval. In this section we present the example use-case of the storage service, We also show how the requirements of the application can be fulfilled by the system based on the NDS architecture deployed in the PLATON infrastructure.

One of the planned applications of the popular backup/-archive service is the storage of the PIONIER network traffic. The operator of this national academic network is forced by the legal regulations to store at least network protocol headers of some percentage of the traffic in the network and be ready to present these data to authorities on legitimate demand.

The application requirements can be summarized as follows. Firstly, the storage needs of the application are around 168 TB per year. This leads to a need for the average data storage performance at the level of 5.5 MB/s. Secondly, the network traffic data are collected in multiple geographical points, i.e. in more than 20 nodes of the PIONIER network. Thirdly, high responsiveness to I/O requests is expected from the storage system as the data are collected in real-time. Fourthly, the physical data replication is required for the data durability, as the operator has to be ready to present the traffic data to authorities during at least 5 years from collection. Fifthly, we can expect that the I/O requests generated by the application in the storage system are mainly related to frequently appending the data to existing traffic log files and rarely creating new log files or circulating the logs. Thus, a relatively small number of I/O operations requires extensive meta-data handling.

The expected level of the data traffic incoming to the system from multiple geographical points allows to use multiple virtualized Access Nodes running on physical machines distributed across all sites of the PLATON infrastructure. This guarantees enough data storage throughput, as multiple Access Nodes will run multiple instances of NDS Virtual File System, as well as high responsiveness of the system to I/O requests, thanks to the geographical proximity of Access Nodes (and associated Storage Nodes) to the point in which the network traffic is collected. The performance of processing I/O requests offered by the NDS instance should be sufficient to store the network logs in a real-time. Multiple virtualized Storage Nodes can be used in order to assure the existence of at least two data replicas, and thus to guarantee the data durability. Finally, meta-data database used for this application can be run on a dedicated virtual machine or DBMS shared with another applications as appending the network traffic logs does not require a lot of meta-data processing.

V. STATE OF THE ART

Storing massive amounts of data for backup or archival reasons in the distributed environment is known to be a challenging problem, and there are a number of systems that attempts to solve it. In this section we shortly present selected systems and show how NDS differs from them.

The dCache project provides a distributed, heterogeneous data storage system visible for the user as a single virtual file system [DCA1]. Depending on the persistency model, the data are automatically exchanged with backend (tertiary) storage systems, replicated, migrated to 'hot-spots' and recovered in case of disk or node failures. Beside its specific protocols, data in dCache can be accessed via NFS, FTP, HTTP, WebDAV and GridFTP. The system design assumes a big number of users accessing the same file system, as the data exchange and scientists and projects cooperation aspects are very important in the dCache application (opposite to the NDS system assumptions).

The meta-data of the dCache system is managed by the Chimera service [CHI1] based on the relational database. These features make dCache similar to NDS; however, some differences follow. It is possible to mount a lower level service as a sub-tree of a higher service, but there is still a central root service. This solution enhances performance, security and availability of the data. The authorization is based on Unix attributes or on certificates and ACLs. The system was designed for grid structures and it has been used mainly for grids so far. On the contrary, NDS was designed as data archive and backup service for institutions that require a high level of confidentiality, so that the logical workspaces and the metadata (as well as physical replicas) for different clients are separated. This allows also for better system distribution, especially thanks to the fact that no common view of the file system (such as common root in dCache case) is needed.

iRODS (Integrated Rule-Oriented Data System) [IRO1] is a software for data-grids, digital libraries, persistent archives and the real-time systems. The data is managed according to a set of rules that define replication modes that are optimal from the point of view of data security, access time, load balancing and possible failure recovery, etc . The system is object-oriented – the stored data can be files, database objects or any other digital data. This was achieved using virtualization techniques on many levels of the system. The system consists of a central meta-catalog (based on relational database), rules engine, executing engine (a set of micro-services that perform single operations) and scheduler. The system was tested by NASA and certified as appropriate for commercial solutions, but it was deployed mainly in the scientific environment. The main differences to NDS are a central meta-catalog and the fact that iRODS is object-oriented. The rules system, while providing great flexibility, makes iRODS relatively difficult to use.

Overall, according to our best knowledge, there is no distributed data storage system that might be used for data backup or archival purposes and would guarantee full scalability of the single system instance. Both dCache and iRODS exploit the meta-data management systems that are logically centralized, similarly to NDS. Note, however, that the architecture of NDS system combined with the features of the PLATON deployment infrastructure including support for servers and storage virtualization, allow to deal with the limitations resulting from meta-data centralization within a single system instance.

VI. SUMMARY

In the paper we have shown the architecture of the National Data Storage system and discussed the possibilities of deploying this architecture in the redundant servers and storage infrastructure of PLATON project. The infrastructure supports advanced features such as servers and storage virtualization which enables implementing scalable, reliable and secure data backup/archival services, designated for short-term data protection and long-term data archival.

The NDS system architecture itself allows for implementing data storage services that address critical user requirements such as high data access performance, huge storage capacity, service reliability and data storage security. The scalability and reliability of the system is achieved, among others, by means of multiple, geographically distributed Access Nodes and Storage Nodes for serving the user's I/O requests and storing the actual user's data. However, the NDS system has a logically central meta-data database, for integrity and performance reasons, and as a result of the end-user requirements related to the data security (most of the potential users demand a logically separate name-space isolated from the other system users).

We have shown that meta-data centralization in the single NDS system instance can constitute service scalability bottle-neck in many aspects. However, we have also presented the idea of deploying multiple instances of the NDS system in the redundant PLATON infrastructure composed of multiple servers and data storage resources supporting the servers and storage virtualization.

Our analysis shows that the NDS system instantiation allows to omit scalability limits resulting from centralized meta-data catalog. Moreover, the usage of virtualization technology along with the NDS system configuration options enables flexible set up of the NDS instances, which can help addressing various, often contradictory requirements of different users of the popular data backup/ archival service, basing on shared or dedicated pools of physical resources.

References

[IDC1] IDC analysis. Cited among others in: Humans created 161 exabytes of data in 2006. http://www.itnews.com.au/News/74870,humans-created-161-exabytes-of-data-in-2006.aspx

[PIO1] PIONIER – Polish Optical Internet – a nationwide broadband optical network for e-science. http://www.pionier.net.pl/online/en/projects/69/PIONIER_Network.html

[NDS1] National Data Storage project in Poland. Project Web page: nds.psnc.pl

[GUS1] Source: polish Central Statistics Office. http://www.stat.gov.pl/gus/index_ENG_HTML.htm

[MIN1] Source: polish Central Statistics Office. http://www.stat.gov.pl/gus/index_ENG_HTML.htm, cited by: http://www.studenckamarka.pl/serwis.php?s=73&pok=1909

[FUSE1] Filesystem in Userspace. http://fuse.sourceforge.net

[PSQ1] PostgreSQL. The world's most advanced open source database. http://www.postgresql.org

[SLO1] Slony-I. Enterprise-level replication system. http://www.slony.info/

[VMW1] VMware vSphere Hypervisor (ESXi). http://www.vmware.com/products/vsphere-hypervisor/index.html

[HSM1] HSM. TSM-HSM, Tivoli Storage Manager for Space Management. http://www-306.ibm.com/software/tivoli/products/storage-mgr-space/

[DCA1] http://www.dcache.org

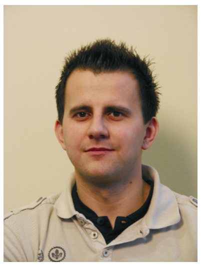
[DCA2] P. Millar, dCache. Presentation during 3rd Terena TF-Storage Meeting, Dublin, 2009. http://www.terena.org/activities/tf-storage/ws5/agenda.html

[DCA3] P. Fuhrmann, V. Gulzow, dCache, storage system for the future, In: W. E. Nagel, W. V. Walter, W. Lehner (Eds.): Euro-Par 2006, Parallel Processing, 12th International Euro-Par Conference, Dresden, Germany, August 28-September 1, 2006, Proceedings. LNCS 4128, Springer 2006,

[CHI1] Chimera – a new, fast, extensible and Grid enabled namespace service, http://www.dcache.org/manuals/chep06/Chimera-paper-chep06.pdf

[IRO1] https://www.irods.org

[IRO2] Arcot Rajasekar, Mike Wan, Reagan Moore, Wayne Schroeder, A Prototype Rule-based Distributed Data Management System HPDC workshop on "Next Generation Distributed Data Management", Paris 2006,



Maciej Brzeźniak, M.Sc. in Computer Science from the Poznań University of Technology (2001). Current affiliation: Supercomputing Department in Poznań Supercomputing and Networking Centre (http://www.psnc.pl) as storage systems specialist and researcher. Interests include: scalable data storage and management technologies and architectures, storage and data management services design, implementation and operation, storage systems performance. Involved in: PLATON project (Service Platform for e-Science, POIG.02.03.00-00-028/08), National Data Storage project (nds.psnc.pl) and TERENA TF-Storage (www.terena.org/activities/tf-storage). Previously active in CoreGrid project (Institute on Knowledge and Data Management, EU FP6 NoE project no. FP6-004265), Atrium project (IST-1999-20675) and national projects: SGIgrid (mathlib.psnc.pl) and Virtual Laboratory (vlab.psnc.pl). His publication record includes papers and technical related to performance analysis of storage and computing systems architectures as well as storage and data management services.

Dr. Norbert Meyer is currently the head of the Supercomputing Department in Poznań Supercomputing and Networking Center (http://www.man.poznan.pl). His research interests concern resource management in GRID environment, GRID accounting, data management, technology of development graphical user interfaces and network security, mainly in the aspects of connecting independent, geographically distant Grid domains. NM conceived the idea of connecting Polish supercomputing centres, vision of dedicated application servers and distributed storage infrastructure. He is the author and co-author of 60+ conference papers and articles in international journals, member of programme committees of international conferences related high performance computing and distributed computing. He was the leader of RINGrid EU project, currently leading the EU DORII project.

Rafal Mikołajczak, M.Sc. degree in Computer Science from the Poznań University of Technology in 1998. Currently he is employed as a deputy manager of Supercomputing Department in Poznań Supercomputing and Networking Center. His research interests concern checkpoint low level services and grid service, distributed data management and data center technical infrastructure. Hi is also responsible for the storage system in PSNC. He was working on the kernel-level checkpointing mechanism for the IA64 with Linux OS and he was working on the Grid Checkpointing Architecture within the CoreGRID project. The other area of interested is the storage systems architecture and technology. Currently he is working on the National Data Store project.

Gracjan Jankowski received the M.Sc. degree in Computer Science from the Poznań University of Technology in 2002. He works in Supercomputing Department of PSNC since 2001. In years 2001-2008 his R&D activity concerns checkpointing, load balancing and migration of running processes in HPC and HTC. He is author and coauthor of a few papers discussing the checkpointing related issues. He was participating in two national projects: PROGRESS (http://progress.psnc.pl) and SGIgrid, implementing kernel and user level checkpointing for Solaris and Linux (checkpointing mechanism for IA64) operating systems – http://checkpointing.psnc.pl/. The projects PROGRESS and SGIgrid were co-funded by the State Committee for Scientific Research and two leading IT companies, respectively SUN Microsystems and Silicon Graphics. Recently he has been involved in FP6 Network of Excellence Core-GRID funded by the European Commission where he worked on techniques allowing utilization of legacy checkpointing packages in GRID environment. Currently he works as analyst and programmer of virtual file system for National Data Storage

Michał Jankowski, M.Sc. degree in Computer Science from the Poznań University of Technology in 1998. Current possition: computer systems analyst at the Supercomputing Department in Poznań Supercomputing and Networking Center. Professional interests: distributed data storage systems, user management, authorization, software design and development. Involved in projects: PLATON (Service Platform for e-Science, POIG.02.03.00-00-028/08), National Data Storage project (nds.psnc.pl) – design, development and deployment of meta-catalog. Previous projects: CoreGrid (Institute on Knowledge and Data Management, EU FP6 NoE project no. FP6-004265), Baltic Grid and BalticGrid-II (www.balticgrid.org, EU 6 FP, contract no.: 026715, EU 7 FP, contract no.: 223807), Clusterix (National CLUSTER of LInuX Systems), SGIgrid (mathlib.psnc.pl) – design, development and deployment of Virtual User System (vus.psnc.pl).